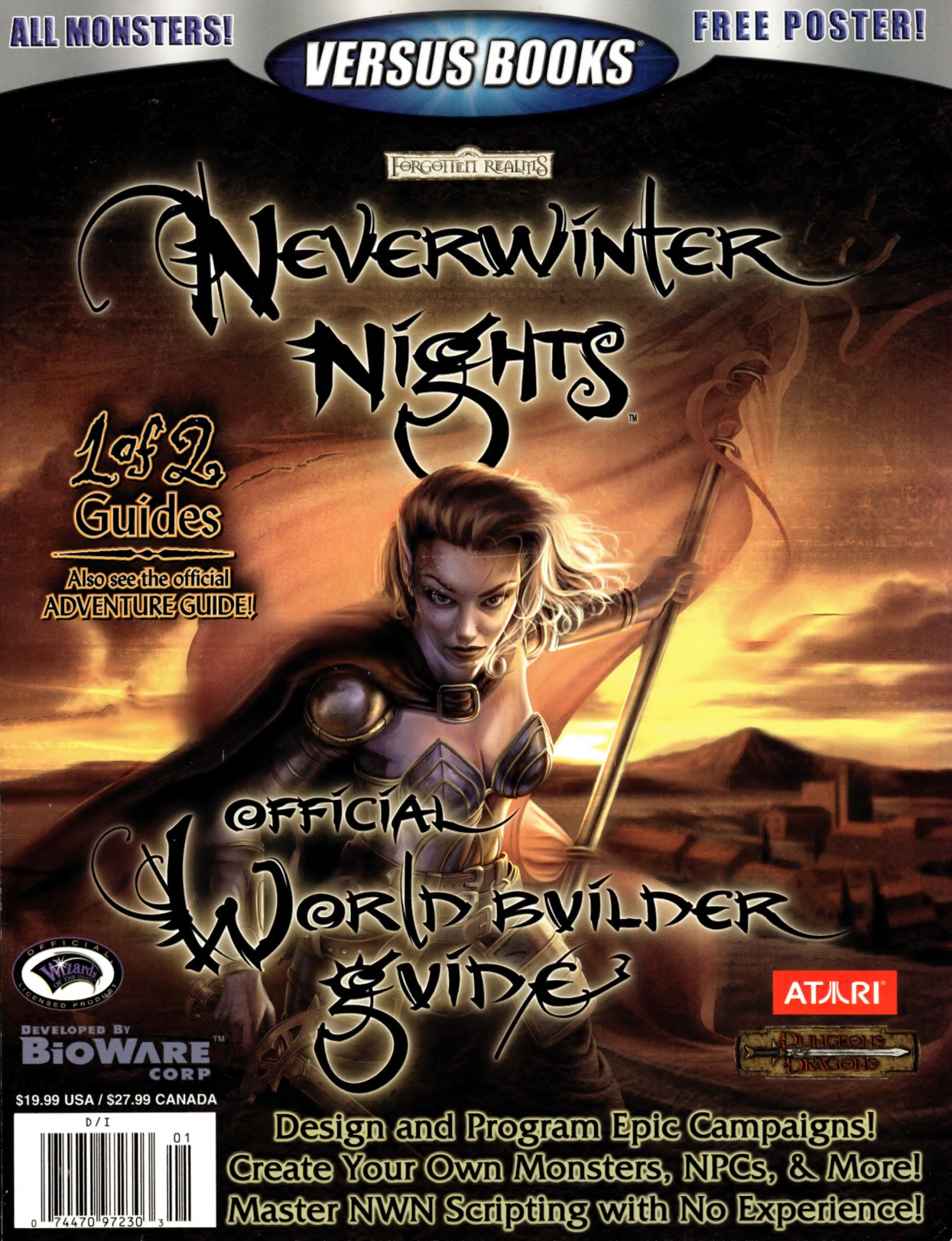
ALL MONSTERS!

VERSUS BOOKS

FREE POSTER!



FORGOTTEN REALMS

NEVERWINTER NIGHTS™

1 of 2 Guides

Also see the official ADVENTURE GUIDE!

OFFICIAL WORLD BUILDER GUIDE

OFFICIAL Wizards of the Coast LICENSED PRODUCT

DEVELOPED BY BIOWARE™ CORP

ATARI®

DUNGEONS & DRAGONS

$19.99 USA / $27.99 CANADA

D/I

01

0 74470 97230 3

Design and Program Epic Campaigns!
Create Your Own Monsters, NPCs, & More!
Master NWN Scripting with No Experience!

VERSUS BOOKS

VOLUME 41

AUTHORS
CASEY LOE
FRANK CROCKETT

ART DESIGN
CRAIG KELLER

LAYOUT
S. LILLIAN SPERRY

ADDITIONAL CONTENT
GERALD GUESS
BRYAN LANE

CEO
MATTHEW TAYLOR

PRESIDENT
LEN CICIRETTO

VP OF EDITORIAL AND ASSOCIATE PUBLISHER
HOWARD GROSSMAN

MANAGING EDITOR
PATRICK CUNNINGHAM

PRODUCTION DIRECTOR
PETER DISCOE

VP OF SALES
SKIP MCFERRAN

CONTROLLER
JANICE RAAB

WEBMASTER
RENEE MALCOLM

ASSISTANT CONTROLLER
PATRICIA LEIGH

SALES ASSISTANT
KAY SMITH

ACCOUNTING ASSISTANT
TONYA ROOPE

ADMINISTRATIVE ASSISTANT
TERRY ADAMS

CORPORATE ADVISOR
PATRICK J. FERRELL

CONSULTING CFO
BOB NETTER

ISBN: 1-931886-13-X

PRINTED IN USA

TELL US WHAT YOU THOUGHT OF THIS BOOK!

E-MAIL: NWN@VERSUSBOOKS.COM
WEB: WWW.VERSUSBOOKS.COM
SNAIL MAIL: NWN C/O VERSUS BOOKS
PMB #404
2532 SANTA CLARA AVE.
ALAMEDA, CA. 94501-4634

NEVERWINTER NIGHTS © 2002 Infogrames Entertainment, S.A. All Rights Reserved. Manufactured and marketed by Infogrames, Inc., New York, NY. Portions © 2002 Bioware Corp. BioWare Aurora Engine copyright 1997-2002 BioWare Corp. All Rights Reserved.BioWare, the BioWare Aurora Engine, and the BioWare Logo are trademarks of BioWare Corp. All Rights Reserved. Neverwinter Nights, Forgotten Realms, the Forgotten Realms logo, Dungeons & Dragons logo, Dungeon Master, D&D, and the Wizards of the Coast logo are trademarks owned by Wizards of the Coast, Inc., a subsidiary of Hasbro, Inc. and are used by Infogrames Entertainment, S.A. under license. All Rights Reserved. Windows and Windows 95/98/2000 are registered trademarks of Microsoft Corporation. All Rights Reserved. The ratings icon is a trademark of the Interactive Digital Software Association. All other trademarks are the property of their respective owners.

ALSO SEE THE *NEVERWINTER NIGHTS ADVENTURE GUIDE*!

© 2002 Empire 21 Publishing, Inc. VERSUS®, VERSUS BOOKS®, and PERFECT GUIDE® are registered trademarks of Empire 21 Publishing, Inc. EMPIRE 21™ is a trademark of Empire 21 Publishing, Inc. May not be reproduced in whole or in part without express written permission from Empire 21 Publishing, Inc.

Empire 21 Publishing has made every effort to determine that the information contained in this book is accurate. However, the publisher makes no warranty, either expressed or implied, as to the accuracy, effectiveness, or completeness of the material in this book.

Special thanks to Rachel Hoagland, Brent Knowles, Ray Muzyka, Trent Oster, Patti Roakes, Lezlie Samuel, Joleen Schafer, Jay Watamaniuk, Mike Webster, and Greg Zeschuk.

NOTE: Due to the ongoing completion process of the *Neverwinter Nights* game and *Aurora* toolset, certain information contained in this guide might be dated or inaccurate at its time of publication. Versus Books has made every effort to ensure that the content is sound, and we apologize for any errors that arise.

The *Dungeons & Dragons® Player's Handbook*, *Dungeon Master's Guide*, and *Monster Manual*, are the ultimate guides to the *Neverwinter Nights* experience. Add the *Forgotten Realms Campaign Setting*, and you can explore beyond the boundaries of *Neverwinter Nights* and into the rest of the phenomenally detailed world of Faerûn. Pick them up at your favorite hobby or book store, and prepare for endless fun. For more information, visit wizards.com/dnd.

World Builder Guide

Welcome to the *Neverwinter Nights World Builder Guide*!

No game-construction toolset has ever given the user as much power and flexibility to create their own games as *Neverwinter Nights* does. The tools are easy to use, but the vast number of customization options can be mind-boggling, and learning the C-based scripting language can be daunting to those with no programming experience.

For this reason, we're going to begin with a basic module design chapter in which we build a short, simple module to illustrate how the toolset is used and what it's capable of. This is just the tip of the iceberg, but we'll be dabbling in all aspects of module creation, from customizing your environments to creating monsters and scripting NPC behavior. At the time of this writing, we don't know exactly what documentation will be shipping with *Neverwinter Nights*. If you've already used a manual walkthrough, in-game tutorial, or module wizard to make a short playable module, and you feel you're comfortable with the way the toolset works, you may want to skip ahead to Chapter 2.

In, Chapter 2, "The Game Plan," we'll show he we planned our own medium-scale Neverwinter Nights campaign, and discuss what level of scripting knowledge is necessary to execute it.

Chapters 3 through 5 are the advanced module design chapters, in which we'll go in-depth into each of the tools, show how they were used to create the official campaign, and apply what we've learned to script specific scenes of our own module.

By the time you hold this completed book in your hands, we hope to have that completed module online for download at both www.versusbooks.com and the official *Neverwinter Nights* community website. We'll do our best to make it an entertaining adventure and to carefully document its creation via comment windows and script notes so you can take it apart in the toolset and see how surprisingly easy it is to bring your visions to life with Neverwinter Nights.

It's been a privilege and an honor to work on this project, and I hope you'll e-mail us at nwn@versusbooks.com with your comments and suggestions.

Happy world-building,

Casey Loe and Frank Crockett

TABLE OF CONTENTS

World Building

Quick Start ... 2
The Game Plan ... 14
The Module and Areas ... 21
Objects, Encounters, and Triggers ... 50
Creatures, Conversations, and the Journal ... 67

Appendices

The C Language ... 106
Monsters ... 126

Preview

Adventure Guide ... 207

Chapter 1: Quick Start

1 Bad Times at the Blackgrove Inn

The forest is dense, rain is pounding, and the road is in poor shape. Nocturnal predators roam the dark woods, but the party presses on, focusing on the hot meals and soft beds said to await in the Blackgrove Inn.

Someone spots a campfire, and the party spots an elderly pair huddling near the entrance of a long abandoned ruins. "You must help us," they plead. "They killed everyone in the Inn. We barely escaped with our lives." You can see the flickering lights of the Inn in the distance, and hear the distant screams of the customers who weren't so lucky.

A little simplistic and maybe a tad trite, but we're here to learn the basics and that's as good a place to start as any. Let's make it happen.

2 Using The Area Editor

Select "Toolset" from the *Neverwinter Nights* main menu, and select "Create a New Module" from the Welcome menu. The **Module Creation Wizard** should pop up automatically. Hit "Next," pick a name for your sample adventure, and continue on to the next window. The "Area Wizard" button will let you pick the size and setting for the first area we'll use in our module.

First, select a scenery **tileset** from the list provided. For this example, we'll start with the Forest. Give it a punchy, memorable name (like "Forest," for example) and hit "Next" to get to the size selection screen.

An area can range from 1x1 to 32x32 and anything in between. Neither the height nor width can exceed 32 tiles, but it's unlikely you'd ever want to. After all, each tile is about 10 scale meters, so a Huge size area (32x32) would equal several city blocks in the real world. That's a lot of turf to build out, populate, and fill with junk. Let's start with something easily manageable, like 8x12 (this size isn't one of the defaults, but you can enter the values directly). Hit "Finish," and then cancel out of the Module Wizard. We can always add more areas later.

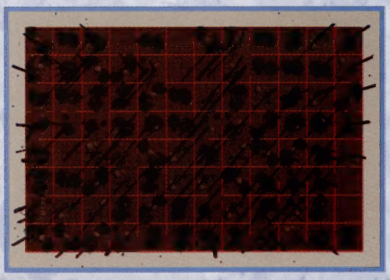
Note

Need a little more space, or have extra to spare? You can always resize your area after the fact. Under the Edit menu, choose Resize Area and enter new values. ***Important: Rows are always deleted from or added to the bottom, and new columns are always deleted from or added to the right. Unfortunately, there is no way to delete from or add to the left or top of an area.***

3 Laying The Terrain

On the right side of the screen you'll see the **Toolbar**. The Terrain Icon is currently selected, and below, in the **Palette,** you'll see a list of the three groups of terrain. Hit the + boxes to open up the three groups, which collectively list all 38 tiles in the Forest Tileset. The nine basic types of terrain are on the bottom, in the **Terrain** group. You'll notice the Toolset has already been kind enough to pepper our forest with trees, rocks, and a carpet of leaves. This is the default Forest terrain. If you don't like a particular tree or rock, or you want to clear-cut the whole area, you can toggle through the various "Forest" titles by clicking on a tile (or click-dragging multiple tiles) and hitting "Delete." You can also paint large amounts of terrain by selecting "Forest" from the Terrain list and clicking it or click-dragging it across the area map.

The Palette



Changing the Camera

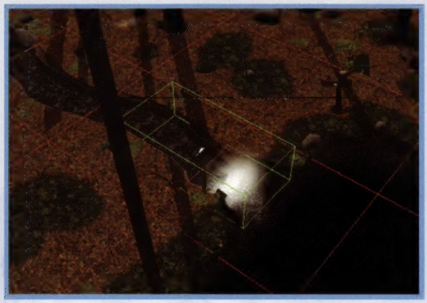
ZOOM: Use the middle mouse slider to zoom in and out of your area. If you don't have one of them new-fangled mouse wheels, you can use the + and – keys on your number pad.

PAN: Hold the Ctrl key and the left mouse button to pan up, down, left and right.

SKEW: Hold the Ctrl key and the right mouse button, and move the mouse left and right to rotate the scene along the X axis. Move the mouse up and down while holding these keys to move it along the Z axis. This is especially useful for replicating the viewpoint of a player in your scene. You can also rotate the scene with the number keys on your keypad.

REORIENT: You can hit the spooky eye button, or "5" on your number pad to snap back to a bird's-eye view at any time.

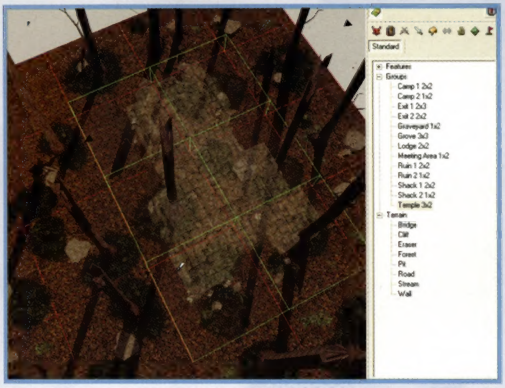
Try picking a different type of terrain, like "Pit," and dragging it across the map. Mix and match the various types of terrain, and watch how the different tiles automatically merge at the edges... A River tile will automatically turn into a bottomless waterfall when it hits the edge of a Pit tile, for example.



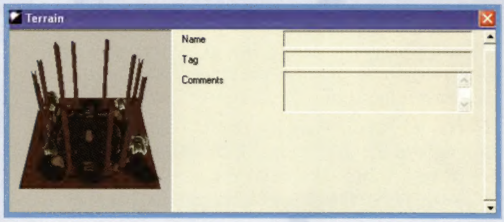
If one tile doesn't automatically replace another (such as when you try to put a road over a pit) you'll need to select "Forest" from the Terrain menu and turn the pit back into basic Forest. Some tiles can also be turned back into basic Forest with the "Eraser" tile.

Features are single tiles that can be put over Terrain. Most of the Features (Big Tree, Camp, Graveyard, etc.) can only be placed on the basic Forest tiles. But some (like Bridges) can only be placed over certain other types of terrain, like Pits or Streams. It's all pretty intuitive, and if you can't figure out what a Feature is for, you can just select it and move the cursor over various types of terrain. When the outline turns green, you've found a match.

Groups work the same way Features do, but they're more than one title wide, up to 3x3.



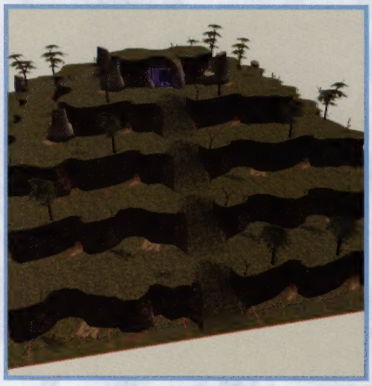
- To rotate a Feature or Group, hit the right mouse button before you place it.
- To get a preview of a Feature or Group, open up the **Show Preview window** by hitting the Show Preview button.



- If you want to clear the canvas after experimenting a bit, click-drag over your map and hit Delete to remove all the Features and Groups. Then pave over your other terrain types with basic Forest.

Adding Depth to Areas

The Rural and City Exterior tilesets have a special **Raise/Lower** tile in their Terrain set. When this is selected, you'll raise the height of a tile with each left click, and lower it with each right click. There is also a **Ramp** tile that can be used to make the higher tiers of these areas accessible to players. Many



of the Features in these areas, such as caves, can only be set into raised areas. There is no limit to the amount of tiers a Rural or City Exterior area can have, but there is no way to add depth to any other tileset.

Viewing Options

There is a variety of check boxes under the Scene pull-down menu.

REFRESH: If you change the lighting or other environmental factors in your scene, select this to update it.

DISPLAY GRID: When you want to admire the beauty of your creation, check this to make that ugly red grid disappear.

DISPLAY SHADOWS: All tiles and objects in your scene automatically cast shadows. Turning this off will make it easier to see what's going on and speed up your refresh rate.

FADE GEOMETRY: This lops off the top part of some tiles, so they won't get in the way when you're placing objects. If you put it on **Object Mode Only**, the scene will get cropped when you're placing objects, creatures, waypoints, etc., but not when you're editing terrain.

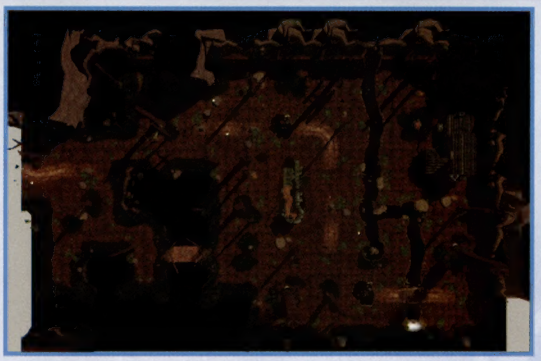
FOG: This will display your chosen atmospheric fog (more on that later). This is nice when you want to see how the final scene will look from a player's perspective, but gets in the way when plotting terrain.

USE AREA LIGHTING: When this is on, the lighting options you've selected from the Customize Environment menu will be displayed. When this is off, it's always high noon on a cloudless day.

Let's Lay Some Forest!

Alright, let's get our feet wet and make a forest. As a general rule, you don't want any openings around the perimeter of the area, unless they're going to lead somewhere (and in this exercise there's nowhere for them to lead to), so let's start by covering the north edge of the map with impassable cliffs and the south edge with bottomless pits. We'll leave a little opening on the west side of the map, which will be our starting point, since the players have to have come from somewhere.

To fill out the needs of our scenario, we'll use a "Lodge" Group in the northeast corner as the Blackgrove Inn, and a "Ruins 2" Group for the area where the players will meet the survivors. Fill in the gaps around the perimeter with more Cliffs and Pits, and then make intermittent use of the Road tile to suggest a road in poor repair. A Bridge across the pit will give players a choice of ways to proceed, and a



Stream near the Inn will add some variety (we'll put in a Bridge for flavor, even though players can walk across streams).

After you've laid the necessary terrain, a few more Features will spice things up. I used the Webbed Forest group, a Graveyard (where the dead comfortably await the afterlife in hammocks) and some Big Trees.

If you want to test it out, hit the Set Starting Point button and place that by the opening on the west side of the map.

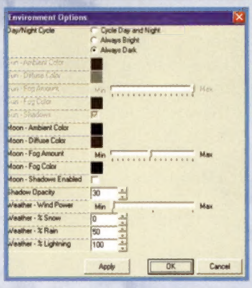
Hold Shift and the right mouse button to rotate it so you'll be facing into the area. Save the module and quit out of the Toolset. Then restart the game, go to "Main," select "Other Module" and choose your sample module from the list. Now you can run around to your heart's content.

Pretty cool, huh? Of course, it's lacking a few things. Our scenario calls for a stormy night, but it's clearly midday, it isn't raining, and that blue haze isn't very spooky. Let's go back to the Toolset and see what we can do about that.

4 Customizing Your Environment

Go to **Area Properties** under the Edit Menu, and click on the **Visual** tab. There is a variety of preset environmental settings here, but it's easy enough to make our own. Pick one that sounds close (say, "Exterior Raining") as a base and hit the "Customize Environment" button. This will bring up the **Environment Options** window.

This nifty window lets you control the Day/Night Cycle, Fog, and Weather. We'll start by clicking the Always Dark button for our Forest, which spares us the trouble of having to set up different fog and color settings for day *and* night.



The **Diffuse Color** is the color of the main light source, which strikes the forest floor and the tops of objects. A bright full moon on a cloudless day would be white, but we'd want a much darker color to make our forest appropriately spooky… We'll go dark grey. The **Ambient Color** is the color of the reflected light that pervades our area. It will have a huge impact on the scene: A dark forest green will do wonders at setting the mood, but it comes at the expense of covering up some of the natural colors in our scene.

If you exit the scene and select "Use Area Lighting," from the Scene menu, you'll see the results. Note how drastically our scene has changed! The Customize Environment option is the key that changes happy elf-settlements into deep woods, well-lit shopping plazas into dank dungeons, and idyllic small towns into dens of death and deceit. Don't worry if our forest looks too dark; we can always add some campfires, and players with torches and spells will have their own light sources as well.

Playing With the Fog Machine

The Fog settings in the Environment Options window are great for adding atmosphere. White or light gray will give the scene a cool misty, ethereal effect, but we're looking for something a little darker. Just because the slider says "Fog" doesn't mean you have to emulate Fog specifically; putting the slider three quarters of the way up and making the Fog color black will give our scene a nice blanket-of-night effect. If you want to check it out, zoom into your scene and then select Fog from the Scene menu.

The last set of Environment options are pretty self-explanatory. You can adjust the intensity of the wind power, and the odds of different types of weather. Crank the rain and lighting to 100%, and our stormy night will be guaranteed.

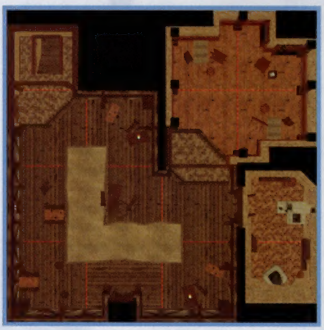
While we're in the neighborhood (of the Area Properties window), hit the Audio tab and pick some appropriate tunes. Now let's save the module, head back into the Main game, and check it out. If it still doesn't feel right, tweak some of the environmental variables and try again.

5 Adding Placeables

When it comes to making realistic, memorable settings, laying terrain is only half the job. The other half is Placeables, a list of objects that you can select by clicking on the table icon in your toolbar.

Like the Terrain options, they come neatly organized in a bunch of different categories. Unlike the Terrain options, there are a bajillion of them.

While our forest has a lot to gain from a few well-placed rocks, shrubs, campfires, and corpses (corpses make *everything* better), we'll switch to our second game area for this example. Relaunch the Area Wizard, and make a new area with the City Interior tileset. The small size (4x4) should be fine, since this is just a small forest inn. Use the Terrain options to make an entrance on the south wall, put a stairway down to a basement somewhere, and mix up the tiles and features to get a few different looking rooms.



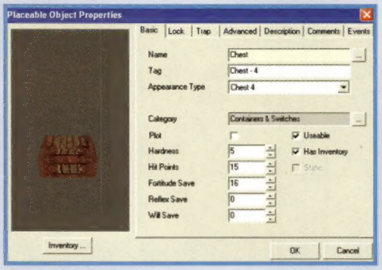
As you can see, each tile comes with a few pieces of furniture. We've got tables and benches in the serving room; tables, chairs, and bookcases in the living room; stoves and the like in the kitchen. But that's not enough… Where will people sleep? Choose Beds from Placeables: Miscellaneous Interior and set a few in the living room.

Sigh… All the bedrooms in* Neverwinter Nights *end up looking like 50's TV sitcom sets—always a pair of twins.

If they're not facing the direction you want, hold Shift and rotate them with the right mouse button. Drop a few other things around the room... Rugs, chests, corpses, book piles, more corpses, etc. Pretty simple, right?

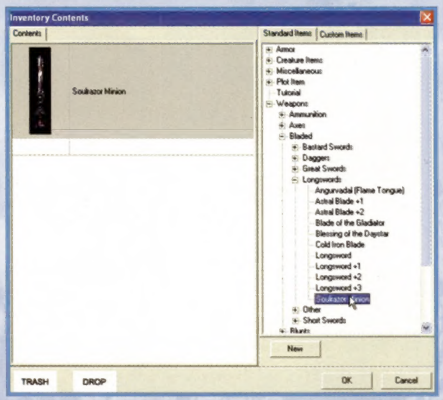
All of the containers in the Containers & Switches category come with treasure included. The contents, which include gold, potions, and occasional weapons, scrolls and armor, are generated randomly by a script referenced in the object when opened or destroyed by a player. The quality of the treasure varies by container; of the four chests, the first has the lowest quality treasure, and the last has the best. This is the same for the corpses, crates, loot bags, etc. The bookcases and bookpiles generate random books and scrolls.

If you want to put something specific in an item, you'll need to edit the object's properties. To do so, select the object, hit the right mouse button, and choose "Properties." The **Placeable Object Properties** window will pop up.



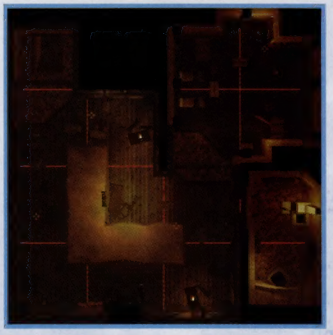
This menu will also allow you to change an object's attributes and description, set locks and traps, and give it scripts and conversations, We'll cover all of that in more depth in Chapter 4.

For now, we'll just fill up the chests and other containers. To put things in an item, the **Useable** and **Has Inventory** boxes in its Basic Properties must be checked. All the items in the Containers section have these checked already. If you want to put an object into something else, like slip a Soulrazer Minion Sword under the innkeeper's pillow, select his bed, go to Properties, and check the Useable Box to make the bed selectable by players. Check Has Inventory to allow it to contain items, and then the Inventory button to bring up a list of treasure. The Standard Items tab lists all the items in the game, and the Custom Items tab includes all the ones you created (none as of yet). Select an item you like, and drag it into the Contents box on the left.



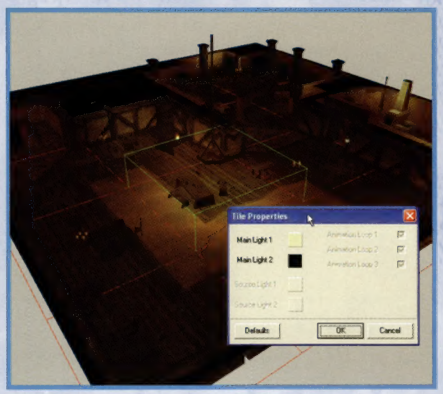
If you want to check out an item's parameters, right-click it and choose Edit Copy to bring up the Item Editor. To get rid of the random treasures, either unclick the Useable box (making the object entirely unselectable) or delete the treasure-generating scripts in the Scripts tab.

Illuminating the Inn



Don't forget to place some light-emitting objects (braziers, candelabras, flames, etc.) and use the Customize Environment option under Area Properties to give your inn the lighting you want.

You can further customize the lighting by right-clicking on a tile with the Paint Terrain button selected, and selecting **Tile Properties**.



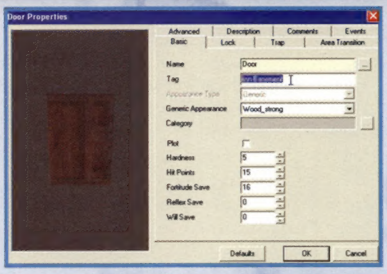
You can change the **Main Light Colors** for each tile (make them black to turn them off) and modify any **Source Lights** (such as the lamps on the tables) in the scene.

Since the Inn has fallen on hard time, we need to add one final touch... A pile of corpses lying atop bloodstain placeables. Now we're cooking with gas!

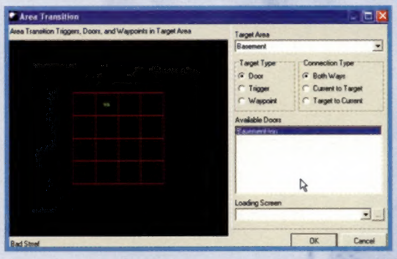
Connecting the Doors

We'll make one more area for our example—the Basement—a 4x4 Mines and Cavern scene with a large cavern and a set of stairs that lead back to the Inn. Well, they don't lead there yet, but they will. First we need to put a door in the doorway. Click on the door button, and pick a door you like from the list selected. Stick one copy of it in the stairway in the Inn, and the other in the stairway in the Basement. Select the Inn door, go to the **Door Properties** windows, and give it a tag in the Door Properties "Basic" tab.

Tagging the one in the Inn "Inn-Basement" and the one in the Cavern as "Basement-Inn" is a good way to keep track.



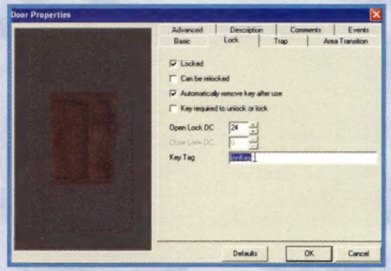
Next, go to the **Area Transition** tab in the door's Property window. Under Destination Type, select "Door." Then hit the **Setup Area Transition** button, and select the Target Area from the list in the Area Transition window.



If you've selected the Inn Area door, select the Basement as your target area. Either choose the door by name or select it by position in the grid map. Make sure "Both Ways" is the **Connection Type**, and hit OK. Set up a Forest-Inn door and an Inn-Forest door, and repeat the process. That's all there is to it–save your module and go from area to area to check it out!

Editing doors is just like editing placeables. Let's say we want to lock the door to the Inn and slip the key under a flower pot. First, right-click on the door and select Properties. We have some decisions to make… Should the key be the only way to unlock the door? If so, select the **Plot** button to make the door indestructible. We want the player to have other options, like picking the lock or bashing it down, so we'll leave it as is. Currently, its **Hardness** is 5 (meaning 5 points of damage will be deleted from every hit) and it has 15 hit points, which seems right. You can adjust its saving throws if you really want to get down to brass tacks, but I'm happy with the defaults.

Click on the Lock tab, and click the **Locked** button. We'll also click **Automatically remove key after use**, so people won't have to carry the key around once it's outlived its usefulness. Since we want the players to have other options, skip **Key required to unlock or lock** and **Can be relocked**, since that won't matter in our module.



The **Open Lock DC** defaults to 18… But since players can always take 20 when opening a lock outside of combat, a DC that low won't keep out even the most inept Rogue. Let's set it at 24 so it'll be easy for a Rogue and possible (but challenging) for any other class. Since we didn't click on **Can be relocked**, we don't need to worry about the **Close lock DC**.

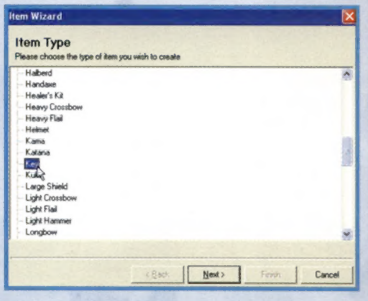
All that's left now is to fit the key to the lock… Type "InnKey" into the **Key Tag** field. Inn Key doesn't exist yet, but we'll make it soon enough.

Note

If you wanted to add a trap, you could do that here too. Click on the Trap tab and check the Is Trapped box. Pick your trap from the Trap Type pulldown menu, select the Detection DC and the Disarm DC (once again, it will usually be a freebie if it's under 20) and keep the defaults on everything else. But since that wouldn't really make sense in the context of our scenario, we'll skip the trap this time.

Crafting the Inn Key

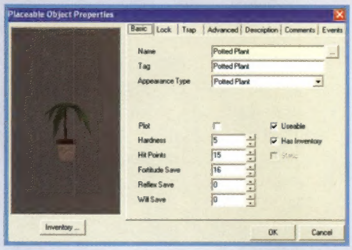
Making your own items is a snap, and keys are no exception. Hit the Item button, right-click in the palette, and choose "New" to bring up the **Item Wizard** (or select it from the Wizards menu).



Select Key from the list, hit Next, and name it "Inn Key." The tag, which is the name the program uses internally, will drop your spaces and end up as "InnKey" (if you chose a different name, make sure to note the tag and re-enter that into the Forest-Inn door's Key Tag field). Next, you'll be asked to choose the palette category where your item will be placed in the Custom Items list. This is for your own personal organization, so put it someplace you'll remember, like Plot Items. Hit Next, check Launch Item Properties, and hit Finish.

In the **Item Properties** window, there isn't much to change. Select **Plot Item** so the key can't be destroyed, and leave the rest of them as they are. Hit the Appearance tab and tweak the design and color of the key's bitmap (as it will appear in your inventory). 3, 5, 3 for the model is nice. Heck, they're all nice. Don't lose a lot of sleep over it.

Grab a Potted Plant placeable from the Miscellaneous Interior folder, put it by the Inn's front door, and edit its Properties.



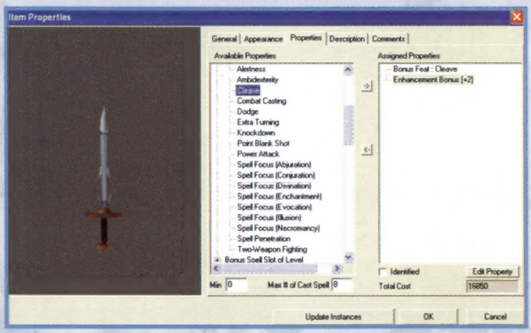
Click on Useable and Has Inventory, and then click on the Inventory button. Select the Custom Items tab, and drag Inn Key into its Contents. Now when players click on the plant, they'll be able to drag the key into the inventory, and the door will open right up. Or, if they don't notice the plant, they can bash it down or pick the lock.

Now for the fun stuff... You can change its Appearance the same way you did with the key, and give it a cool description. Under Description, the **Unidentified Description** is what players will see before they successfully identify it, and the **Identified Description** will appear once they do. If you don't want your item to require Identification, go to the General tab and click on **Identified**. You can also jack up the price here (but not reduce it) and set the number of charges for wands and the like.

When you're done crafting your dream item, make a chest, put it in the Basement, and delete its default treasure-generating scripts. Hit the Inventory button under the graphic, and drag a copy of your new weapon from the Custom Items list to the Contents tab.

6 Weaponsmithing, Made Easy

While we're making items, let's try creating something a little more exciting, like a weapon. Bring up the **Item Wizard** again, and pick something fun, like a Long Sword. Give it an appropriate name, and put it in the Weapons: Bladed: Longsword folder. Pull up the **Item Properties** Window and prepare to go to town. Select the Properties tab, and you'll see a list of all the various attributes that can be assigned to your item. When you see one you like, double-click on it, and it will be moved to the **Assigned Properties** list. To edit the property, double-click on the listing from Assigned Properties, and you'll be able to adjust the **Cost Parameter Values**, like the amount of an Ability Bonus or Damage Bonus, and the **Parameter 1 Values**, like Damage Bonus Type. If you make a change to either, hit the appropriate Select Bonus and then OK. As you add properties, the **Cost**, **Required Level**, and **Required Lore** will automatically be recalculated. There's no maximum amount of properties, but if you get greedy you'll end up with a weapon too powerful to ever be wielded.



7 Adding Random Encounters

Well, it's a lovely little nature walk we've designed, but it would be ever so slightly more compelling if we actually added something to *do*. A few enemy encounters would be a good place to start. There are two ways of adding enemies to your module: as preset creatures, or scalable encounters. We'll add a few of each.

Preset creatures are great when you have a specific battle experience in mind for your players. But when you're just throwing a few random battles at your players, it's easier and more convenient to just use an encounter. Encounters have two parts: A trigger, and a spawn point (or points). When the players step on the trigger, monsters chosen from a selected list appear from the spawn point. There are three reasons you would use an encounter instead of a preset enemy:

- You want the difficulty of the encounter to be scaled to the level of the players.
- You don't want the program wasting time calculating the behavior of dozens of monsters scattered all over a large area that can't even be seen by a player.
- You want the enemies to surprise the players by appearing suddenly.

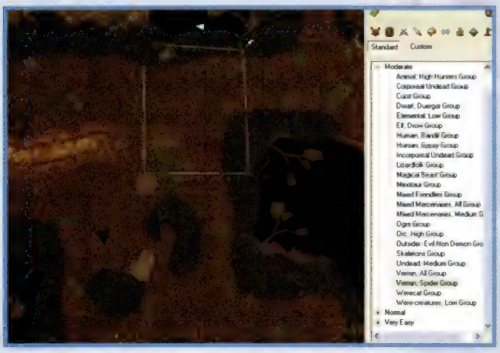
Most of the non-boss enemies in the official campaign are spawned by encounters, usually for the first two reasons.

Spiders, Spiders, Everywhere

The spider webs in that 'Webbed Forest" tile are calling out for a nice spider battle. Click on the Paint Encounters button and pick your difficulty level.

The difficulty of encounters is scaled to match the level of the players, but within limits (if an encounter was open to everyone it would have to include everything from Goblins to Ancient Red Dragons). They're divided into five categories. We'll make this adventure for characters level 4-8 (since you've probably played the official campaign a little bit by now and have a character in that range), which is "Moderate" range. There we go... "Vermin, Spider Group" sounds like a winner.

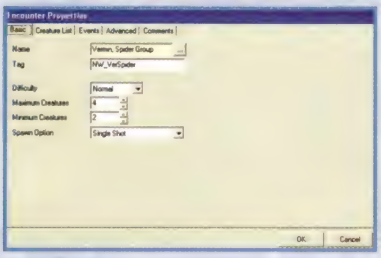
Double-click on that, and move your mouse over the main window. Your cursor will now be a brush, which you use to paint the trigger. Pick a spot a tile or two in front of the webbed trees, and click-draw a trigger box of any shape. Double-click on the last point to close it.



Right-click on your trigger and choose "Add Spawn Point," which will make a big grey arrow appear inside the box. This arrow is where the enemies will appear, so you'll want to select it and move it over to the webs. If you want the enemies to appear from multiple locations, you can create additional spots and put them in different locations.

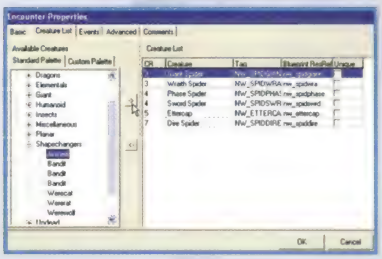
Encounter Properties

So what, exactly, is happening here? If you want more control over your encounter, right-click on it and choose Properties to bring up the **Encounter Properties** window.

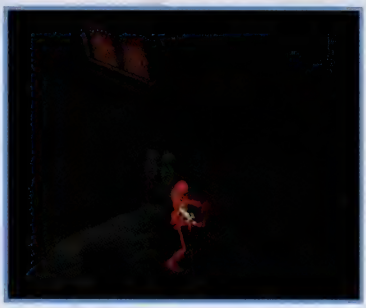


The **Difficulty** here is relative; if you want this encounter to challenge your players, raise it to Hard, and it will examine the players' level and respond accordingly. For a Level 3 character, it will spawn a few CR 4 enemies. For a Level 10 character, it will spawn a bunch of CR 8 enemies. If you set it to Easy, it will instead spawn in a smaller amount of weaker enemies for each character.

The Minimum and Maximum of 2 to 4 sounds about right... You wouldn't want more spiders than that. Hit the **Creature List** tab to see what exactly we're dealing with. The encounter won't spawn all of these enemies, of course—just 2 to 4 of them. If there's a spider that rubs you the wrong way, select it and hit the left arrow button to remove it from the list. If you want to add something, like an Aranea (a shape-changing spider that isn't on the list), select it from the Standard Palette and hit the right arrow button to add it to the list. Each of these encounters will activate only once, unless you set the encounter to respawn under the **Advanced** tab.



All set! We'll add in a few more encounters elsewhere on the map... Some Undead near the bridge, and a visit from the Animal, High Hunters Group near the river. Then save your module and give it a shot. If you find the challenge setting is wrong, it's too easy to avoid the triggers, or the enemies are spawning in too close, you may want to go back and tweak them a bit.



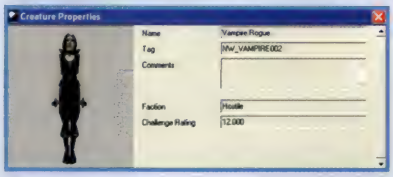
A pair of encounters catch up with our hero at the door to the Inn.

8 Adding Preset Encounters

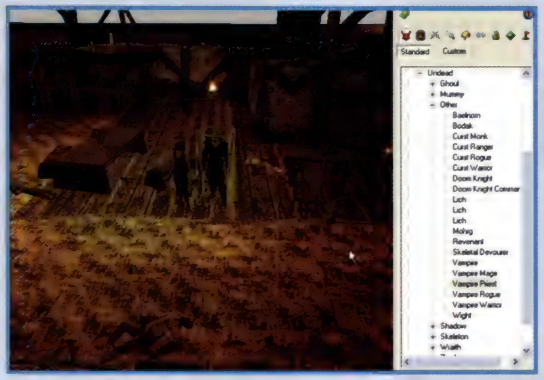
Let's head back to the Inn and figure out what it was that sent our elderly couple running for their lives. Since this is the boss encounter of our little module, it deserves a little more time and energy than the random encounters did. We want something that looks cool, acts cool, and fits the needs of our adventure.

Click on the Paint Creatures button and search through the menus for an appropriate monster (you can also look through the beastiary in the back of this book for inspiration).

How about a cadre of vampires? You'll find them listed under Undead: Other. To take a look at our options, hit the Show Preview Window and scroll through the vampires.



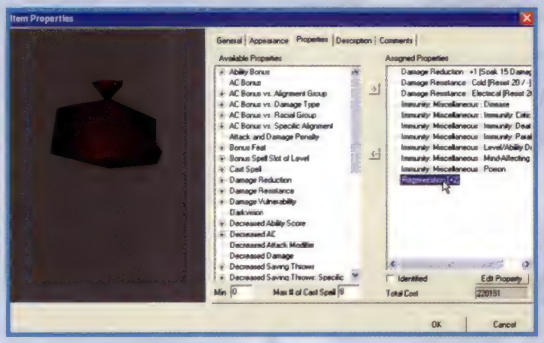
There are four versions, but only two different vampires, so let's make it a couple: The oddly alluring Vampire Rogue and her Vampire Warrior sire. Just click on them one by one, place them in the Inn, and turn them so they're facing the door.



All done! Or… Maybe not. Can a character of level 4 to 8 really handle a pair of CR 12 and 13 Vampires? Maybe we should take our young lovers down a notch. Right-click on one of our vampires (in the Main window or Module Contents window – we want to edit *our* Vampires, not the Vampire template), and choose Properties. You can give them names in the Basic tag, then switch to the Classes tab and lower their levels to 6 or so. Then hit the Statistics tab, and take the stats down a bit. Is a Natural AC of 6 really necessary? And 74 base hit points for a mage? Try 0 and 45, respectively.

Next stop is the **Feats**, **Spells**, and **Special Abilities** tabs to make sure she isn't packing anything too dangerous. Clearly, some of her spells are way too powerful… Scroll through the list, and if you see anything problematic (like Cloudkill or Stoneskin) put the number memorized to 0.

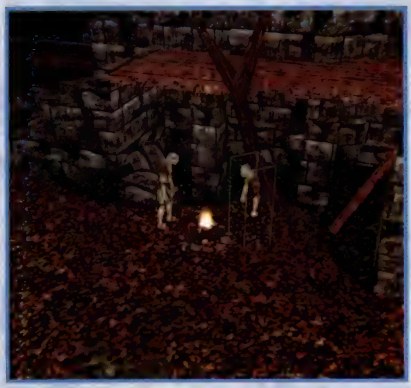
Hit the **Inventory** button, and let's see what she's carrying. A Dagger +1 isn't bad, but what about her **Natural Equipment**? She's packing a Vampire Bite as her Special Attack (natch) and wearing a Skin/Hide of Vampire Properties. The Skin/Hide is where all the powerful Defensive abilities of many creatures hide, and Vampires are no exception. Right-click on that to bring it up in the Item Properties window. Hmmm… Maybe we should lower the Regeneration a bit? Let's make it +2. Note that we just made a module-wide change, permanently altering the properties of the Vampire Hide. Since these are the only vampires we'll be using, it doesn't matter, but in general it would be a better idea to save it as a Custom Skin/Hide and equip it to our Vamps manually.



Cripple the Vampire Warrior in a similar fashion, and this encounter should now be possible for mid-level characters.

9 NPCs and Conversations

NPCs, like the elderly couple that asks our party for help, are created in the exact same way. Hit Paint Creatures, and open the NPCs folder. Choose an Old Man and Old Woman from the Humans folder and place them near our forest ruins, where they'll be visible to players. They're looking a little rough, so let's give them a nice campfire placeable to warm those old bones.



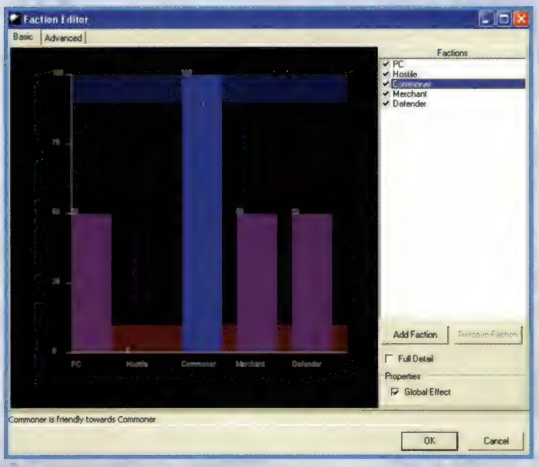
You can edit them in the same way you edited the vampires, but since they won't be involved in combat we're concerned with a different set of stats. They'll need names, for example, which we'll entrust to the random name generator (hit the * button by the name field until you get one you like). I ended up with Salino and Nelie... Not too shabby! You can hit the "..." to the right of the portrait to change their default images, and further change their appearance by dressing them in different clothes in the Inventory screen. (Note that we can't use the options in the Appearance tab, because we're using a template instead of creating our own).

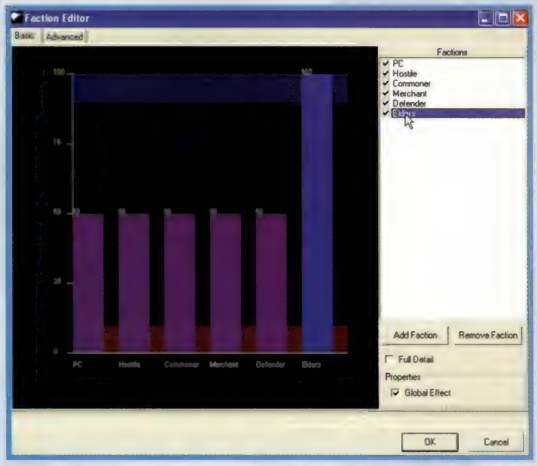
10 A Brief Introduction to Factions

Since we have NPCs in an area with monsters, we have to be concerned about what will happen if they run into each other. There are Spiders right near the ruins, and we don't want grandpa and grandma being killed before they can talk to the players! One way to deal with this problem is with **factions**, the attribute that determine how NPCs feel about other NPCs. Currently, the monsters are all in the Hostile faction (as all monsters are) and our elderly couple is in the Commoner faction. Go to the Advanced tab, and hit Edit Factions to see what that means. Click Commoner in the Factions List to bring up their graph... It seems Commoners like other Commoners a great deal (their friendliness is at 100), are neutral to PCs, Merchants, and Defenders (friendliness 50), and are hostile to members of the Hostile faction (friendliness 0).

Members of the Hostile faction are fond of each other, but hostile to everyone else. That's a problem, because commoner Salino and the hostile Spiders are going to fight.

Fortunately, the **Faction Editor** provides an easy solution. Hit the Add Faction button and name the new faction "elders". Since they look out for each other, raise the friendliness of elders to other elders





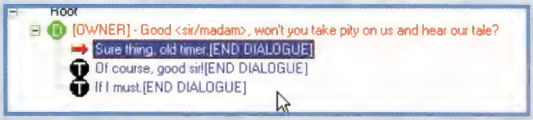
to 100. Put everything else at 50. Then go to the Hostile Faction and put their opinion of the elders to 50. Now the elders are indifferent to hostile creatures, and vice versa.

Hit OK, and make sure grandpa and grandma are both assigned the Elder Faction before we proceed.

Since grandpa and grandma are important characters, we may also want to hit the "Plot" button in the Advanced tab. That will prevent them from being killed by anything, including malicious PCs and collateral damage from misfired Cloudkills.

11 The Conversation Editor

Bring up the Conversation Editor from the Tool menu, and let's script a conversation for Salino. This is a lot easier than it looks. Just hit the Add button in the upper left-hand corner, and it will prompt you for the first dialogue of the NPC. Input something appropriate, and hit OK. Then select that line, and hit the Add button again to write up a player response. Keep hitting the Add button on the original "Owner's line (the Owner being Salino in this case) if you want the PC to have different responses to choose from.


Root
[OWNER] - Good <sir/madam>, won't you take pity on us and hear our tale?
Sure thing, old timer.[END DIALOGUE]
Of course, good sir![END DIALOGUE]
If I must.[END DIALOGUE]

Note the <sir\|madam> in Salino's opening line. This is called a token. The software will automatically check the player's gender and say the appropriate thing. To bring up a list of tokens to add to conversation, right-click in the Enter Dialogue window and choose Insert Token.

There will be times when you won't want what the NPC says to change based on the player's response. In those cases, continue the conversation from just one of the responses, and then right-click on the first line of the continuation, choose Copy, and select one of the other responses. Right-click on that, and choose "Paste as Link." Ta da! The text will appear in grey, and when the conversation gets to it, it will go right back to the original string. Mess around with some of the conversations in the official campaign if you're unclear how this works.

Adding animation to your conversation is a cinch. Select the line where Salino is talking about how his inn was taken over, then click on the **Other Actions** tab in the lower right-hand corner. Scroll through the animations and choose "Talk Forceful." Uncheck the **Loop** button so he'll only do it once. When he thanks you for agreeing to help, go back to that tab and choose a Bow animation.

To assess your conversation, right-click on the first line, and choose Test. The test function is crude, but it should get the job done. Save your conversation, and then input it in the Conversation field under the Basic tab of Salino's character properties. Make a conversation file for his wife, and you'll be all finished here.

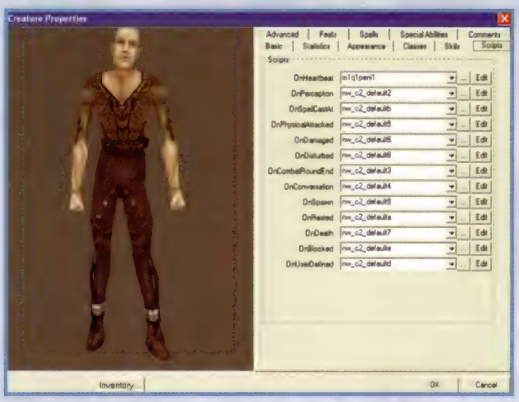
12 Dabbling in Scripting

To those of us who don't know C (or any computer programming languages for that matter), the word "scripting" can be pretty terrifying. But I want to show you how easy it is to harness some of the power of scripted events without knowing anything about scripting at all. Our module is pretty much done… We have our areas, our encounters, our characters, our boss, and our boss treasure. If there's too much combat, or not enough, you should know how to tweak that. If the bosses are too hard, you know how to tone them down. If you want more treasure, you know how to add it.

But there are a few things in the execution that just seem crude. Why are Salino and Nelie just standing there stiffly? Why do they keep repeating the same thing over and over even after you've agreed to help? Why do the bosses have so little personality? Couldn't they taunt you or something? Shouldn't Salino acknowledge when you've completed the quest? These are all scripting issues, and they're easier to solve than you may think. By the end of this book, you should know how to implement all of the solutions. But since it's pretty easy stuff, we can solve most of these problems right now, before you even know *how* to script.

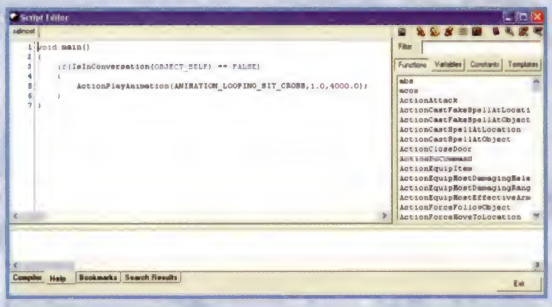
One way is by scavenging scripts from the official campaign. Remember how the displaced citizens of Neverwinter were sitting cross-legged outside of the gates that led to their home districts? If we can figure out what makes them sit, we can steal that script and put it in our sample module! Open up Chapter 1 of the official campaign and find the City Core. Scroll around until you find a character who was sitting in the game. You'll note they're not sitting now; since the sitting is caused by a script, it won't "fire" until you load up the game.

The character named Harris is one of the sitters. Right-click and bring up his Properties, and select the **Scripts** tab. Here is where you'll find all of the scripts that determine how Harris behaves. On the left are the names of the "script handlers." The script handler is the condition that makes the script activate. For example, the script to the right of OnDamaged does nothing until Harris is dealt damage.



All of the scripts are defaults, except one… And since Harris' behavior in the game is unusual in only one way, that's probably the one we're looking for. Hit the Edit button next to the script in the OnHeartbeat script handler. That will bring up the **Script Editor**.

Even without any scripting experience, you can probably tell what this script does… The first part makes him say something. The second part causes him to perform the cross-legged sitting animation when he's not in conversation. Exactly what we were looking for.



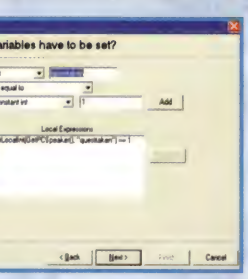
You can find the script in the Module Contents window and Export it to our sample module, or just copy the text of the script. Then go to our module, open up Salino's Scripts tab, and either load in the script or paste it over his current OnHeartbeat script. We don't want the first part of this script, which makes him talk to himself, so delete lines 3-6. (After reading our Appendix C, you'll know enough about the scripting grammar to know the "void main()" and brackets must be in every script). Save it as a different filename, and the editor will automatically compile it for you. Give Nelie the same script in her OnHeartbeat window, and then check out the results.

The moral of this story? You don't need to know a lot about scripting to rip off scripts from the official campaign and use them in your own.

Now let's fix the conversation bug... You talk to Salino, he asks you to help him, and you agree to, but what happens when you talk to him again? He says the exact same thing! But you already agreed to help! That's just stupid.

You could scavenge from the official campaign to deal with this problem, but using the handy-dandy Script Wizard built into the Conversation Editor is an even easier solution. Open up Salino's conversation, and add another node to the Root, that says something like "Have you had any luck? The gods be with you, brave adventurer!" Go to his first line of conversation, the "Good <sir/madam>" one, and hit the **Text Appears When...** tab in the corner. Hit the Script Wizard button, and then check the **Local Variable** box. Keep all the defaults. This variable will be an int (integer). Input the name "questtaken". In the last box, input "0." Then hit Add. Save the script as Salino1.

A local variable is something you create, which the software will remember when the conversation is over. Think of it as a numerical switch: Right now it's off (at 0). When you accept the quest, it will be flipped on (to 1). At the beginning of your conversation, Salino's conversation will check to see where the switch is, and react appropriately. So for the new line, we want to do the exact same thing... Create a script that looks for the local variable "questtaken," but this time, it checks to see if it's 1. Save this as Salino2. Now we need one more script, to flip the switch after you accept the quest. Go to Salino's line after you accept the quest, and select the **Actions Taken** tab this time. Open the Script Wizard, and you'll have three new options. Choose ***Set* Local Variables**, leave the defaults, and add "questtaken" and "1" to the blanks. Add it, and save it as Salino3. Now Salino will remember that you've already agreed to help, and say only the second line.



Since we're only dabbling in scripting now, we won't cover the other stuff here. But these same techniques we used above, mixed with a little creativity, can solve the rest of the problems we listed just as easily.

13 The Post-Mortem on Our Sample Module

This simple module may not be a masterpiece of modern game design, but it's pretty amazing what one can whip up in an hour or two. This exercise was just a quick introduction to the basic features of the *Neverwinter Nights* toolset. If all you aspire to create is combat-oriented dungeon crawls, then you already know just about everything you need to. But if you'd like to create a complicated adventure with vivid settings, memorable characters, scripted events, custom content, and interactive storylines, then read on... We've barely scratched the surface of the *Neverwinter Nights* toolset.

Chapter 2: The Game Plan

1 Crafting Bigger and Better Things

If your dream is to make a digital version of a favorite real-life *Dungeons & Dragons* campaign, you'll need to be willing to make a great many compromises. Not only does the Toolset have its limitations, but the medium of computer games does as well. For example, I've been a player in *D&D* campaigns where the players have become so swept up in political intrigue and elaborate plot twists that we've played for days without any combat, and never missed it. But when I'm playing an RPG on the computer, I get bored with "town stuff" and start itching for action after about 15 minutes. So when you look for ideas and inspiration for your campaign, similar computer games may be a better frame of reference than your real-life role-playing experiences.

That said, your adventure need not be accessible to everyone, and there will be a huge demand for modules of all varieties. You'll find an eager audience for everything from simple dungeon crawls to intricate roman à clefs and ambitious persistent online worlds. We're going to try to make a module that falls somewhere in the middle: A quest for 1 to 4 players with a strong central objective and a fairly linear storyline with a few subplots and side quests. This is our first time making a game, so the final product probably won't be a masterpiece (I hope you'll try it out and let us know what you think), but it should serve as a decent example of how to craft, script, and DM (dungeon master) a moderately ambitious module.

2 Making Your Design Document

All games begin with a design document. For professionally released games, I've seen design documents that range from 100 pages or so to dozens of packed binders. Since this is just for your personal use, you can be as organized (or unorganized) as you want to be, but I strongly urge you to make a rough skeleton for your module and stick with it. If you design as you go, you'll have a nightmare when you try to piece it all together, and that lack of organization will be evident to the players. The design document for our module ended up being about 20 pages, and although we revised it a million times, it was an invaluable framework. It included (in order of creation):

- The basic idea, backstory, and setting of our campaign
- A few character bios
- A flow chart or outline of game events
- A list of game areas
- A rough overworld map
- A list of scripts we'd need
- A list of major conversations we'd need
- A list of custom content we planned to create

The overworld map didn't have much detail; it was just a piece of very fine graph paper in which each square represented one tile. We sketched out the whole area, then divided it into eight rectangular game areas. That way everything fit together nicely, and we had a reminder that if one area had a river as its east barrier, we'd need a river as the west barrier to the area east of it. I would advise against mapping out your towns and dungeons until you're very familiar with the tilesets; otherwise you'll have trouble matching the available tiles to your vision.

Our list of areas and scripts gave us a checklist of things to do, and a way to gauge the scope of the project. That way, when I sat down at my computer every night, I could pick an interesting area to paint, conversation to write, or script to puzzle through, and check it off the list. The lists also gave me a way to divide the tasks into those that did and didn't look like fun, and assign everything in the latter category to my co-author—the poor dope. This is why if you're working with a group of people, it's important that *you* hold onto the gameplan.

3 Coming up with a Good Story

Rather than convert an existing campaign, we decided to create a new one. With enough time and energy, you can do just about anything you want, but life is short and we're going to make it easy on ourselves by choosing a campaign without a lot of bustling cities, with a fairly linear plot, and which utilizes the tilesets, placeables, and monsters provided.

If you lack inspiration for your module, you can base it on an existing *D&D* campaign, convert a book you like, or flesh out a subplot of the official campaign. *Dungeons & Dragons* sourcebooks like the *Dungeon Master's Guide*, the *Forgotten Realms Campaign Setting* and even the *Monster Manual* are full of great ideas; we picked an area out of the

Forgotten Realms so we would have an easy jumping-off point for our campaign and so we could justify buying hundreds of dollars worth of sourcebooks on the company credit card. Um…but that last reason may not be so relevant for you.

4 Our Idea

Two small frontier settlements, cut off from the rest of civilization, are battling nature and each other to eke out a living in an unforgiving part of the world. As if they didn't have enough problems, they've just been discovered by a xenophobic native race that wants them gone at all costs. The PCs were hired to escort a supply caravan to the region, but its unexpected destruction at the hands of the enemy has left the PCs stranded in this difficult world. As the only neutral party in the region, the PCs represent the only hope of destroying the enemy and saving the region.

Not much of an idea yet, but from small acorns mighty oaks do grow. Or moderately sickly oaks, in our case. Hey, it was our first try.

We wanted to do something with a frontier motif, since that would involve a few human settlements but wouldn't require us to make any large cities. It was also fertile ground for human-on-human and human-on-monster conflicts, and would explain the preponderance of monsters in the area, which is good for your backstory. (It bugs me when you start in a huge town in a well-developed area, but the cave a few hundred yards out of town is still full of monsters and there are mysterious ruins a mile away that no one's fully explored. Wouldn't *someone* have gotten around to those things in the last few hundred years? But enough about me and my petty pet peeves.)

Starting with a spot in the Forgotten Realms (between Tethyr and the Cloven Mountains), we fleshed out the pair of settlements and wrote their backstories (see "The Backstory" sidebar). Putting your players in the story is always difficult, especially if you want your module to be open to any class and any alignment. A caravan isn't blisteringly original, but any *D&D* class could be useful as a caravan escort, and it sure beats meeting in a bar.

The Backstory

This text will appear in a book titled The Caravanner's Atlas of Faêrun, found amidst the ruins of the caravan in the opening scene of the game.

History

The hills east of Tethyr are desolate and unforgiving. Nearly inaccessible and overrun with vile beasts, banishment to the Cloven Foothills was for many decades a death sentence for the most wretched criminals of Darromar. Whenever execution was not an option, usually due to fears of inflaming riots in minority communities, the magistrates of Tethyr sent their prisoners through the heavily guarded Cloven Pass, confident they would never return.

Eventually, enough of the banished survived to form a small settlement, which they named Wretch's Haven. Caravans tried their luck at locating the settlement and attempting to barter for local wares, but this was a risky enterprise with little potential for profit. Or so it was thought, until the day when guards at the pass searched a returning caravan and found that the palettes of dried fish contained a number of well-concealed nuggets of the world's most precious metal. Pytheas Blackenshield, a dwarven miner exiled to Wretch's Haven for a brutal drunken murder, had struck mythril.

When word reached Darromar, the crown immediately claimed dominion over the Cloven Foothills and sold the exclusive mining rights to the influential duke Einhander. The dwarven mining company that the duke hired never found Blackenshield's secret mine, but they did locate two small mythril veins near the Greyslough River. There, the duke established a mining colony called Rivenvein.

Nial Einhander, the duke's nephew, was put on the throne of Rivenvein and charged with transforming it from a frontier settlement to the greatest city east of Darromar. Unfortunately, mythril is as elusive as it is profitable, and the Rivenvein colony has yet to find enough to turn a profit for the duke.

Meanwhile, the people of Haven have dropped the "Wretch's" from their land's name, and have transformed their anarchic backwater into a thriving city with a powerful despot and a strong sense of community and order. Though the Einhander family is furious that Haven continues to illegally mine the king's mythril, Rivenvein is not strong enough to wipe out its better-established neighbor. Haven, meanwhile, dares not risk the wrath of the crown by going to war with Rivenvein. So for the last decade, the two have shared an icy truce.

COMMERCIAL PROSPECTS

Taking a caravan to the Cloven Foothills is a very risky enterprise that is possible only in late summer, when the pass is open. You will need to hire several powerful escorts to deal with the hostile ogres, trolls, and giants that populate the area, and a round trip expedition could take more than four months. There is almost no communication between the colonies and the established world, so a caravan might risk life and limb only to find that there has been little or no mythril mined recently. But with Tymora's blessing, you could turn a huge profit if a fresh mythril vein had been found that year.

The people of Rivenvein and Haven are self-sufficient, and the great number of dwarves in the population ensures that their weapons and tools are of masterwork quality. However, many items are nearly impossible to find in the foothills and are in high demand. Past caravans have managed to trade textiles, dried fruit, grain, alchemical supplies, books, bows, musical instruments, and even exotic magical reagents for raw mythril.

Make no mistake—this is an expedition only for the very brave, very foolish, or very desperate. A few caravan companies have been saved from bankruptcy by a successful trip to the Cloven Foothills, but just as many have gone bust in the attempt, and a few have never returned.

5 The Plot and Task Lists

We plan to put in optional tasks for players to complete on their journey, but the progression of our main plot is linear. That makes it easy to summarize, as shown here:

SCENE 1

Caravan Wreck (rural)
Yuan-Ti Raiders' Cave
Various Dead End Caves

The player returns to the smoldering debris of the caravan he or she was hired to guard. The caravan was attacked by an army of monsters led by a few Yuan-Ti raiders. The Yuan-Ti took most of the carvanners as slaves and led them into a cave in the nearby hills. The player can rescue a few survivors.

On the trail of the Yuan-Ti, the player will witness Grisst, a Half-Orc barbarian and fellow caravan escort, battling the raiders' minions. The odds will be slightly against the barbarian, so the player will need to intervene to save Grisst. If he does, Grisst will insist that they continue into the caves to save your employers, and will offer to join your group as a henchman.

The raiders' trail leads to an impassable underground lake. A caravan member, badly wounded and left to die by the Yuan-Ti, can be found at the shore. He sends you east to Rivenvein to get help.

SCENE 2

Rivenvein Rural
Rivenvein Urban
Various Houses and Shops

The caravan's destination was Rivenvein, due east of the wreckage. The wooded hills gradually become farmland, but there is evidence of recent carnage, such as human corpses in the crops. A farmhouse has been looted, but if a player goes into the basement, he can battle a moderately strong enemy for treasure.

Eventually the player crosses the bridge into Rivenvein, revealing a city in ruin. Most of the buildings are destroyed and powerful monsters—both native beasts and infernal fiends summoned by the Yuan-Ti—have taken over the ones that are intact. There are no apparent survivors, although the castle is locked and intact.

Past the castle, toward the north gate that is the only exit, a Haven Scout battles local enemies. The scout will tell you that Yuan-Ti invaders who summoned infernal fiends within the town walls destroyed Rivenvein. Some of the survivors are in Haven now. He is not forthcoming about his business in the ruins. He suggests that you visit Haven to get assistance for the caravanners. Later, he might become a henchman (class: Ranger).

SCENE 3

Various Transition Areas
Haven Rural
Haven Urban
Various Houses and Shops

After Rivenvein, the player can follow the road to Haven or take detours into other areas to explore. The player is allowed into the heavily guarded Haven and sent to speak with the town's ruler, Lord Neven.

Lord Neven offers to send his scouts to escort your caravan companions back to Haven for healing, and explains that the Yuan-Ti were likely behind the raids. In exchange for this favor, he would like to hire your services. The refugees have told him the rumor that only a few days before the city was destroyed, a caravan from Darromar brought a delivery from the queen of Tethyr to the duke of Rivenvein. The gift was a magical weapon designed to destroy the Yuan-Ti. No one is sure what it is or how it works, but the refugees believe it still lies in the castle vault. Neven's scouts are survivors of Rivenvein, living in the tunnels beneath the city—but they are hostile to Havenites, believing them to be behind the destruction of the city. Neven hopes they will be neutral to the player, and hires you to investigate the veracity of the rumors. A refugee gives the player a key to one of the tunnels. (Some of the survivors do not blame Haven and have sought refuge there.)

There are many side quests available in Haven (for example, a Haven alchemist hires you to get items from the wreckage of his competitor's shop in Rivenvein). Shopping and henchmen are also available.

SCENE 4

Rivenvein Urban
Underground Tunnels

Rivenvein is unchanged, although the player can now access previously inaccessible areas if he has accepted various subplots. The door to the tunnels in one of the houses can now be unlocked, but first the player must defeat the monsters that are squatting there.

In the tunnels, the player meets the survivors of Rivenvein. The player is told that the rumor is true. In fact, Rivenvein received two deliveries from Darromar in recent years, both designed by the queen's personal mage. One is a series of ward stones that had long protected Rivenvein from magic being cast into it (summoning hostile creations within the town walls was a favorite tactic of the Yuan-Ti). The more recent delivery was a mysterious weapon designed to wipe out the Yuan-Ti. To test it, Duke Einhander temporarily deactivated the ward stones. At that moment, they were attacked in full force by the Yuan-Ti summoners. The survivors believe that spies in Haven tipped off the Yuan-Ti about the ward stone's deactivation, since it was widely known that Haven spies had infiltrated Rivenvein.

Despite their hatred of Haven, the survivors hate the Yuan-Ti even more and are willing to help. But the weapon remains locked in the castle vault. The only key to the vault is the magical amulet the duke wears around his neck. Unfortunately, the duke was taken captive by the Yuan-Ti. The survivors do not know where the Yuan-Ti base is. They suggest that you ask Neven. He must know how to find them, since they believe he conspired with the Yuan-Ti to destroy Rivenvein.

SCENE 5

Haven Urban

When the player returns, Neven denies the charges of betraying Rivenvein, but does indeed know where the Yuan-Ti are. The Yuan-Ti conflict began when Haven miners, searching for more mythril, unknowingly bored a hole into a Yuan-Ti underground temple. The miners were slaughtered, and the Yuan-Ti were alarmed to discover they had visitors in the foothills. Since then, they've single-mindedly sought the destruction of the surface human settlements. That mining tunnel still exists, and his scouts have confirmed that the Yuan-Ti are using it and other entrances to stage their raids. The Yuan-Ti are much too strong to be defeated, even by Neven, but perhaps a small party could slip in, recover the duke's amulet, and escape alive.

SCENE 6

Transition Areas (Forest)
Fire Giant's Run (Rural)
Mine Shaft

The entrance to the mine is north of Haven, in a maze of cliffs known as Fire Giant's Run. There are many detours and random encounters on the way. A few Yuan-Ti sentries and their minions guard the mine shaft, but they can be defeated in direct combat (difficult) or cleverly evaded. Eventually, the player finds the place where the shaft breached the temple walls.

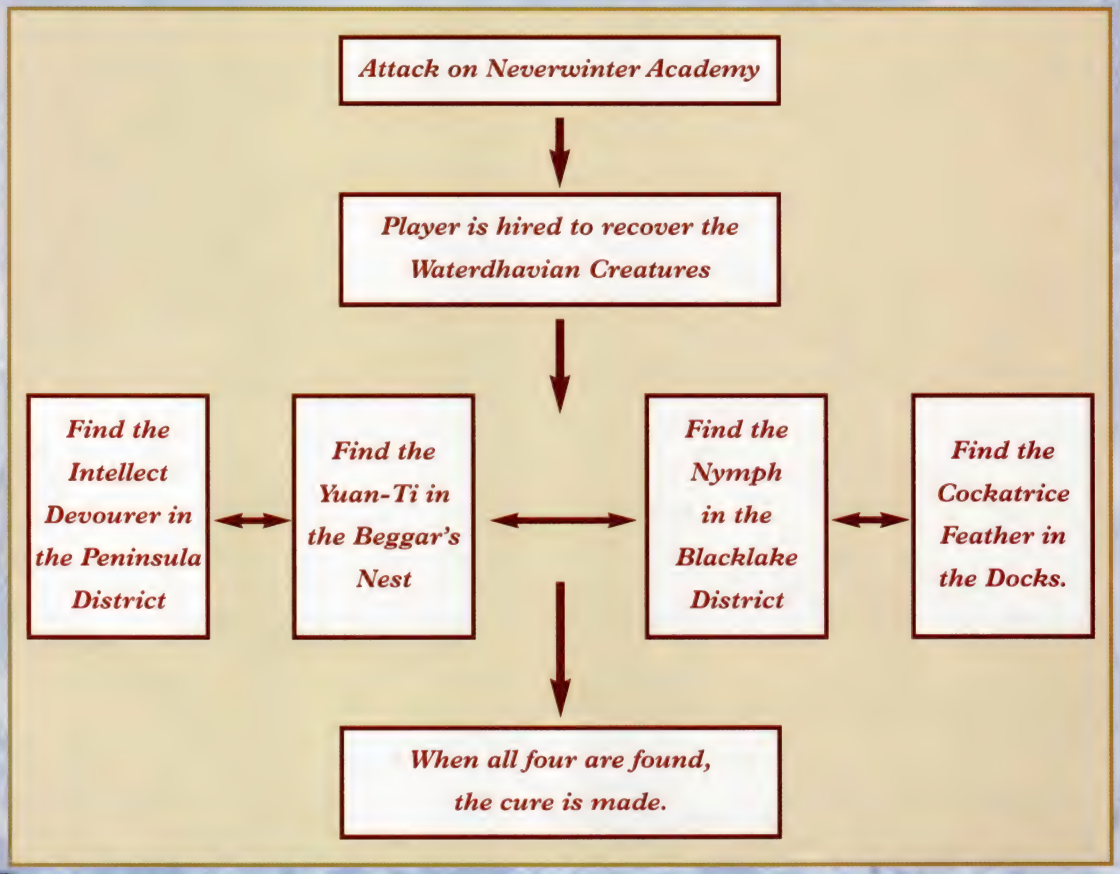
SCENE 7

Yuan-Ti Temple (Crypt)
This is the module's longest and most intricate dungeon. Here, all aspects of the Yuan-Ti plan can be viewed—from the horrible vivisection research on the survivors of Rivenvein (and the player's caravan) to reports from the Yuan-Ti spy that infiltrated Rivenvein and was behind the well-timed attack on the city. (This report can be taken back to Rivenvein and shown to key characters. The final page of the report reveals that the spy has now infiltrated Haven and will report on the right time to strike! As a subplot, the player can try to discover the Yuan-Ti spy amidst the Rivenvein refugees in Haven, based on the clues in the report.)

The duke is dead, but his amulet can be recovered. All of the survivors are now beyond saving, and many of them have been converted into twisted undead.

(To be continued in the module)

The official campaign in *Neverwinter Nights* offers an example of a more flexible story progression. If you choose to go this route with your module, you may find that a flow chart is a better way of organizing your campaign. For example, the main quests of Chapter 1 of the official campaign would look a little something like this:


Attack on Neverwinter Academy
Player is hired to recover the Waterdhavian Creatures
Find the Intellect Devourer in the Peninsula District
Find the Yuan-Ti in the Beggar's Nest
Find the Nymph in the Blacklake District
Find the Cockatrice Feather in the Docks.
When all four are found, the cure is made.

Next, we go through our outline and come up with a list of tasks. We have 13 primary areas to create (counting all the little cave tunnels and houses, we'll end up with 30 or 40 areas in the module). We have several scripted events in a few of those areas, and we'd better make sure we have at least a rough idea of how to do each of them before we go any farther. If there's a script you can't figure out how to make, find a workaround or cut it out now. No matter how good you are, you'll have to cut, add, and modify some things to make your general game vision work as a *Neverwinter Nights* module. And it's a *lot* easier to make those changes now than it will be after you've devoted hours to crafting them.

Example Script List

SCRIPTS FOR AREA 1 ("Caravan Wreck")

- Grisst must be programmed as a henchman. (We'll steal the henchman scripts from the official campaign.)
- NPC caravan members captured by the Yuan-Ti should be spawned in sitting down (ANIMATION_LOOPING_SIT_CROSS). Players can free them by unlocking their manacles (by a skill check within the conversation script), at which point they rise.
- Frightened NPCs will flee from the player following preset waypoints, on an OnPerceived script.
- After following the enemy to the edge of the underground lake and speaking to the last captive, a variable is set that marks the end of this quest. When the player returns to the Caravan Wreck area, this variable is checked by an Area onEnter script, and results in:

1. The surviving NPCs being jumped to the remains of the caravan, where they have set up camp
2. All the enemy encounters and remaining enemies being destroyed
3. Different conversation strings with the surviving NPCs

CONVERSATIONS IN AREA 1 ("Caravan Wreck")

- Captured Caravan Leader
- Escaped Caravanner
- Grisst
- Dying Caravanner in Caves

We'll add plenty more scripts as we think of new puzzles, interesting characters, and colorful events, but the scripts we listed in our design document are the minimum we need to tell the story we want. Dividing your scripts like this is a good way to assess how much work it will be to script each aspect of your campaign. In my original plan, I wanted it to be possible for the players to plot with the survivors of Rivenvein to kill Lord Neven. But after seeing how many more scripts and conversations it would require to make an alternate route that most players wouldn't take, we scrapped it.

6 Do You Really Need to Know Scripting?

We don't necessarily need to script everything we listed. There are two ways of adding complicated quests, puzzles, conversations, and events to your campaign. One is to script them. The other is to DM them. For example, say your campaign calls for a guardian who won't let you through a locked door until you bring him a note from the king and slip it under the door (say, add it to a mat in front of the door that Has Inventory). If you wanted to script this, you would need to:

- Make a moderately complex conversation for the door guardian and the king, with a few small embedded scripts so the characters know what point of the "Get Note" quest you're at, and they respond accordingly.
- Make a script for the mat by the door that scans any objects added to its inventory and opens the door if it's the right one.

Neither are difficult to do, but for a novice module builder it's a few solid hours of work that will require a bit of research and play-testing.

However, if you're DMing the module, you don't need to do anything except quickly make the relevant objects in the Toolset (the door, the mat, the guardian, the king, and the note). When the players first talk to the Guardian, you possess him and make him say what you want him to. When players talk to the king, you possess him, then simply give the note to the players. When they place the note under the mat, you observe and use your DM omniscience to unlock the door—no scripting or long conversations required.

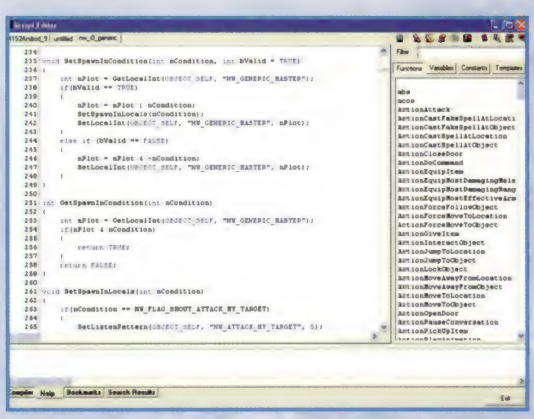
For our sample module, we'll be doing mainly the scripting route. It would be fun to DM the module once or twice, but I'd like it to live on long after I've lost interest in DMing it. And when I am DMing it, I would hate for the module to grind to a halt at a crucial point because the pizza delivery guy is at the

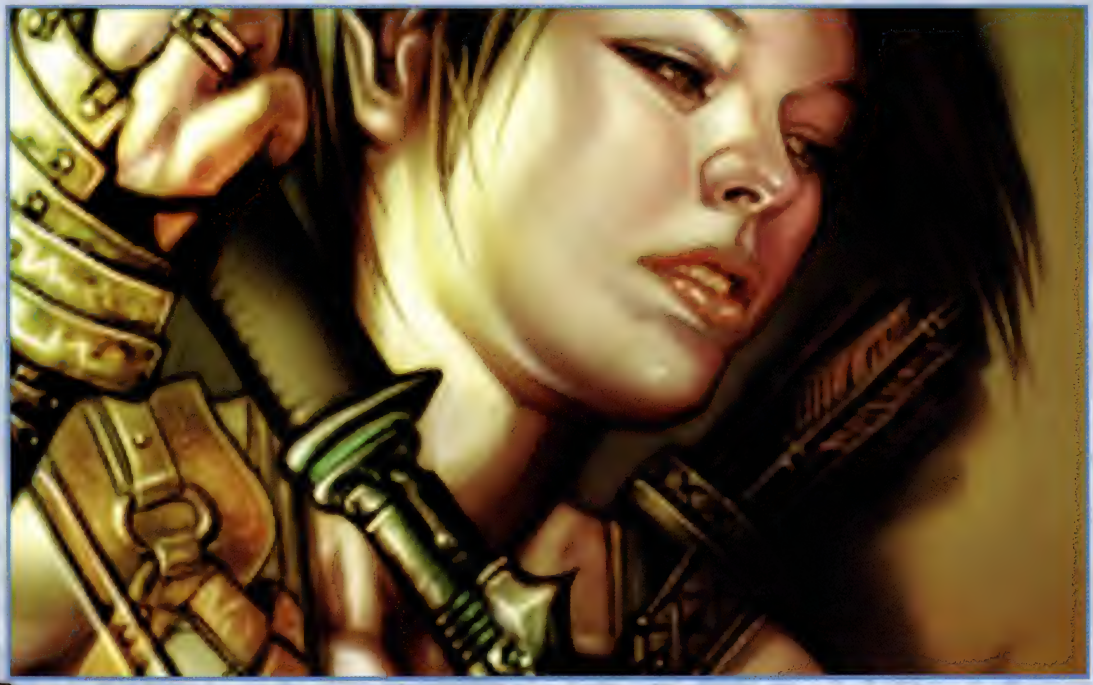
door. Instead, we'll script all of the basic mechanics so the module can be played without a DM, and I'll use my time and energy as a DM to add flair to the characters and make new storyline options available to the PCs (for example, adding the ability to turn against Lord Neven that I had to cut previously). But do remember that it is possible to make very complicated, in-depth campaigns with no scripting if the DM is willing to act out the parts.

7 So You've Decided to Learn Scripting

If you already know the C programming language, picking up scripting will be a snap. So, I want all of you who already know C to take your superior brains and your good jobs and your fancy cars and your many beautiful lovers, and skip ahead to Chapter 3. Not that I'm *bitter*, or anything. It's just that when I was younger, I *wanted* to be a computer programmer. Long story short, I ended up getting some worthless degree in the humanities, and now I spend my life playing video games and plotting the death of my co-author, Frank, who *did* learn C and is now a bajillionaire who runs Microsoft or something.

Where am I going with this? My point is that I am a complete moron, so if your brain implodes when you see statements like "scripting is easy to do…All you need to do is learn C," I understand. Frank has written the scripting sections and **Appendix "C"** of this book for *you*, my friend. You may not have mastered C by the end of it, but it will teach you how to read C scripts and understand what they do. If you spend a couple of days playing with this book and the scripts used in the official campaign, you'll know how to find a model script, implement it, modify it, and make it do what you want when the time comes. That may not sound impressive, but this scavenger style of scripting can get you just about anything your module needs. By the time you start your second module, you may find it quicker to write your scripts from scratch. So go now, my left-brained brothers and sisters, and learns the basics of C. We'll all meet back at the beginning of Chapter 3 to put our newfound scripting skills to use.





Chapter 3: The Module and Areas

1 Module Properties

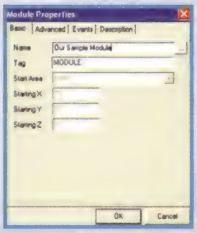
The basic structure of a *Neverwinter Nights* (NWN) **campaign** is made up of one or more **modules** linked together by a common storyline. Each module contains one or more **areas**. These areas represent small sections of the module area, such as the interior of a dark, musty cave; a far-flung border town; or the bustling town square of Neverwinter.

There are plenty of decisions to be made for individual areas, but there are also key decisions involved in making a module-wide level. For example:

- What will the flow of time be in our module?
- How much XP will players be awarded?
- What will happen when a player dies?

These are among the game parameters controlled by the **Module Properties** window, in the Edit Menu. Let's get started!

Basic Properties



Name: The name of your module. Hit the "..." button to bring up a window that allows you to enter different names for different languages. Or better yet, forget all those weirdo Europeans and leave the rest of the languages blank (our apologies to any European readers).

Tag: The name of the module used within the code of the module by certain functions. When you're referring to the module with script functions, you may need to call for this tag.

Start Area: You don't need to enter the area or coordinate values, they're altered whenever you change the location of the one and only Starting Point in your module. To change these values, just click on the Place Starting Point button and put it someplace new.

Advanced Properties

Minutes/Hour: You can alter the length of your module day by changing the number of minutes in an hour. The default of 2 minutes per hour is what the game uses, but there's no reason you can't have your game run in real-time (60 minutes per hour) or even slower (up to 240 minutes per hour).

Dawn/Dusk Start Hour: Change the time that day and night begins. If your campaign features the extra long nights of an *Icewind Dale™* winter, you can set Dawn at 12 and Dusk at 16 (4 p.m.).

Starting Month: The month is pretty much only for flavor, as it can only be seen in journals or used as a token in NPC conversations (they say the number, so "6", not "June").

Starting Day/Hour: As above. If you don't make any changes, the game will begin on June 1st, at 1 p.m. For our *Haven* module, we'll want the starting hour to be midnight (0) so it's nice and dark.

ResRef: The filename of your module, which may be called by certain scripting functions.

XP Scale: Aha! Proof that you've been cheated out of obscene amounts of XP in the official campaign! Yep, you're getting only 10% of the XP for defeating monsters that you should get by 3rd Edition Rules. Since there's a lot more combat in D&D video games than

About Our Scripting Coverage

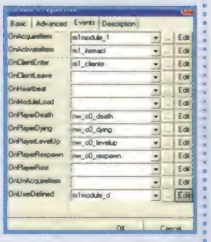
As with everything in *NWN*, modules and areas can be scripted with unique effects, events, behaviors, and play mechanics. In most cases, the script examples you will see are taken directly from the official campaign or our module. This means that the entire script is usually used to illustrate the point. However, in some cases, I will be using only those parts of a script that are related to the current concept you are learning. I will point this out as these instances occur.

All *NWN* scripts are written using the C computer language. If you're experienced with C (or any other language for that matter) it will be easy for you to understand existing scripts, and write new ones. However, for those of you with no experience with the C language, it is a very daunting task to understand or write these scripts. Therefore, we suggest that you skip right to the back of the book and study *Appendix "C": C Scripting and Neverwinter Nights*. It is a valuable discussion of common features you will find in *NWN* scripts. This scripting section (and others in the book) will assume that you can read and understand the examples, given enough time.

in a typical pencil & paper module, it makes sense to keep this low. But if your campaign is short on action, you may want to raise this a bit. If you want to cheat in the official campaign, load up a chapter, jack this up to 200 and save it under a different name. You'll be level 20 in a few hours!

HAK File ResRef: If you make any custom content for your module—new enemy skins, new character portraits, new music, new placeables you made in 3D Studio Max, etc.—you'd put it all in a HAK file, and refer to that file here.

2 Module Scripting (Events)



A module represents the entire collection of areas in your adventure. The **Module Properties Events** tab provides a great many event handlers (also referred to as "script hooks") for module scripts that allow the designer to control some of the highest mechanics of the campaign. For instance, there is an entire group of handlers for determining what happens when a Player Character (PC) falls unconscious, dies at the hand of an Orc, respawns inside the module, and rests during the campaign.

There are 12 handlers for a module. They can be grouped as follows:

Item-related scripts – called when the PC picks up, drops, or activates any item in the module. Useful for scripting quest items and their related behaviors.

Client-related scripts – called when the client (a fancy word for your PC) enters or exits the module. Useful for setting up the initial properties of the module or customizing the module behavior depending on the entering PC.

Heartbeat script – called every 6 seconds (of game time). Not by accident does this coincide with the length of a typical combat round. Useful for providing animations for NPCs in the game, checking on the status of a trap or object, or anything else that requires frequent maintenance. This handler is also available to all other programmable objects in the game.

Module load script – called when the module initially loads. This is before you are able to control your PC. Useful for initializing local variables and setting up initial conditions in the module or the areas of the module.

Player-related scripts – called when the PC is dead or dying (0 or less HP), respawning in the module, and resting during the campaign. A very powerful script group that allows complete control over the PC death cycle, respawn, and resting. More on this later.

User-Defined script – Called by the **SignalEvent ( )** scripting function. Useful for implementing any custom behavior you can imagine. This handler is also available to all other programmable objects in the game.

Scripting For Items

An item is anything in the game the PC is able to place in their inventory. Items can include weapons, scrolls, quest-related objects, and books (to name a few). You may need to use scripting for items when:

- The item develops or completes a quest. For example, a character tells you to find a scroll, and when you find it, your journal is updated.
- The item has an ability that is too complicated to be created by the Item Editor, such as instantly teleporting the PC to a certain area when activated.

Let's focus on the quest-related items first.

Triggering Quests With A Special Item

Commonly, a PC receives a quest by either talking with an NPC or finding some item in the game that is important in some way. For instance, as a module developer you could place an ancient tome in an abandoned dungeon. You could then write a script that would allow access to a secret area when the book is returned to a certain bookshelf or crate. You would want to trigger this quest when the PC finds the book. You do this by creating a script for the **OnAcquireItem** handler, checking each item picked up by the PC for the ancient tome. When the PC picks up the ancient tome, your script updates the journal of the PC with the quest (perhaps requiring the PC to return the tome to the ancient library for a reward). Because the **OnAcquireItem** hook is called every time an item is placed in the inventory, your script just checks every new item against a table of quest items previously placed in the module. If a match is made, the journal is updated and the PC has a new purpose in life! If no match is made, the game progresses normally.

You can also do something special whenever the PC drops (or loses) an item. The **OnUnAcquireItem** script hook is called every time an item leaves the inventory for any reason. For example, you could write a script that triggers a special encounter if the PC tries to drop the book somewhere. Perhaps a vengeful Balor librarian appears and slaughters the PC for not returning the book on time.

Example OnAcquireItem Script

The following **OnAcquireItem** script is taken from Chapter 1 of the *NWN* campaign and illustrates this point nicely. For the purposes of illustration, I have only taken a small portion of the actual script. However, I kept the basic structure intact.

```
#include "NW_I0_Plot"

void main( )

{

  object oItem = GetModuleItemAcquired( );

  string sItemTag = GetTag(oItem);

  object oPC = GetItemPossessor(oItem);


  if (GetIsPC(oPC))

  {


    int iTangleState = GetLocalInt(oPC, "NW_JOURNAL_ENTRYm1q2_Tanglebrook");

/*  additional code removed for retrieving the states of other quests  */


// 1. Update the player's journal entry when they acquire the key to the

// Tanglebrook Estate in m1q2.

    if (sItemTag == "m1q2_KeyTangle")

    {

      // If the player has already received the journal entry from talking

      // with the maid or butler, set the JournalState to 30.

      if (iTangleState == 10)

      {

        AddJournalQuestEntry("m1q2_Tanglebrook", 30, oPC);

      }

      // If the player has not received the journal entry from the maid or

      // butler, use the more generic JournalState of 20.

      else

      {

        AddJournalQuestEntry("m1q2_Tanglebrook", 20, oPC);

      }

    }
```

In the first section of the script, a file (called **NW_I0_Plot**) is included, containing helper functions needed by this script. Then three variables are declared and initialized with the newly acquired item, the tag of the item, and the PC carrying the item. These three items are important because they determine which quest is triggered and whose journal is updated.

The PC object is checked to make sure it is valid and if so, a giant **if-then** statement is entered. All current quest states are retrieved (only the Tanglebrook state is shown in the code), and then the Item tag (**m1q2_KeyTangle**) is compared to each possible quest, looking for a match. In the interests of brevity, I am assuming that the newly acquired item was the Tanglebrook key. Therefore, the first comparison is a match. Because the **if-then** statement (**if (sItemTag == "m1q2_KeyTangle")**) is true, the code inside is executed.

The state of the quest is checked, to see if the PC has already talked with the maid or butler at this point. If so, the journal is updated with a call to **AddJournalQuestEntry( ),** and the **JournalState** is set to 30.

Note

Journal entries and their states are discussed in more detail in Chapter 5.

However, if the PC has not encountered the maid or butler, a more general entry is added, with a different **JournalState**. With the update complete, the script exits, to be activated the next time an item is acquired. Pretty slick, huh? The script is acting as a filter for all found items. Only in special cases is anything unusual done. In all other cases, nothing happens and the script exits without any modifications to the journal.

As mentioned before, a companion script could be written that checks all items dropped or lost by the PC. This script could trigger a special event in such a case; perhaps updating the related quest entry with text indicating that completion is no longer possible or warning the PC that the item is required in later game play. As you will quickly realize, the possibilities are endless.

Scripting Item Activation

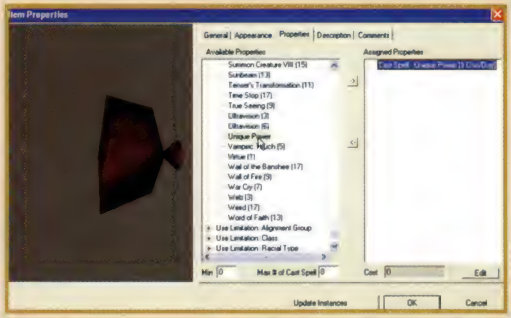
The final item-related event handler is **OnActivateItem**. This event is triggered when the PC activates an item in their inventory with special abilities or powers. A perfect example is the Recall Stone given to you in Chapter 1 of the *NWN* campaign. The Recall Stone is an item that can be activated. If activated, it instantly transports the PC (and any companions) to a secure area. This effect is caused by a script found in the **OnActivateItem** script hook of Chapter 1. Using the default settings of the Script Editor, this script isn't very helpful because both the include file and the **IsRecall( )** function can't be easily found. To find the definition of this function, see "What's Under the Hood?". For your information, **IsRecall( )** was written by the Bioware team and returns the PC and any companions safely to the Hall of Justice in Neverwinter.

However, you could write a script that performs several different actions depending on the item being activated. It would resemble the **OnAcquireItem** script closely, executing different code based on the activated item.

Example: Guenwhyvar

Drizzt Do'Urden is perhaps the most famous character in the Forgotten Realms. This Dark Elf Ranger journeys throughout Faerûn with a few trusted companions, including his magic panther, Guenwhyvar. Guenwhyvar is summoned by a magic item, a panther-shaped figurine that can be used once every other day. So how would you implement Guen (or a similar item) in your campaign? With a simple **OnActivateItem** script.

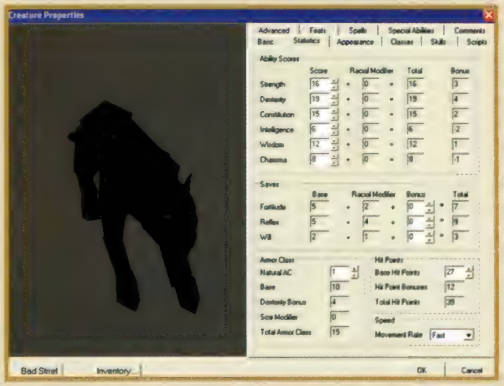
First, we need to set a few things up. We need to make a Custom Item and a Guenwhyvar NPC. Bring up the **Item Wizard** and make a new item of type Gem. Name it "Guenwhyvar Gem" and put it into the Special category. Then launch the **Item Editor**. Change the Stack Size to 1, then go to the Properties tab. Give it **Cast Spell: Unique Power**, open the **Select Property Parameters** window (by double-clicking on the assigned property), and select 1 Use/Day (which is as close to every other day as we can get in *NWN*). "Unique Power" is an important ability that makes fancy custom items like this one possible.



Note

We're using a Gem because there is, alas, no "Panther Figurine" category (there is a Miscellaneous category which has various images of plot items used in the official campaign). Feel free to make your own panther figure bitmap and add it to your HAK File.

Now we need to make Guenwhyvar herself. Bring up the **Creature Wizard** and make a creature with Class: Animal, Level 6. Make her Female, and select "Cat, Panther" for Appearance, and the appropriate portrait. Name her Guenwhyvar, put her in the Summoned Monsters category, and launch the **Creature Editor**. Make sure to change her Faction from "Hostile" to "Defender" (under the Advanced tab) or Drizzt will get a nasty surprise the next time he summons his good "friend." Gwen's actual stats (from the Forgotten Realms sourcebook) are listed here.



GUENWHYVAR			
STATS	CR 5	HP 39	AC 15
	STR 16	DEX 19	CON 15
	INT 6	WIS 12	CHA 8
ALIGNMENT			
Neutral			
FEATS			
Low-Light Vision Weapon Finesse Weapons Specialization (Creature)			

GUENWHYVAR'S ONACTIVATEITEM SCRIPT

When the NPC is done, it's time to script the behavior of the item. As mentioned before, the script would be placed in the **OnActivateItem** event handler of your module. This allows the script to be activated from any area in the game, once the item is acquired. The following script assumes that you've named her Guenwhyvar.

```
void main()
{
   object oCreated;

   location lLoc= GetItemActivatedTargetLocation();
   oCreated = CreateObject(OBJECT_TYPE_CREATURE,"Guenwhyvar",lLoc, TRUE);
}
```

The script above is a basic one that first retrieves the target location of the activation point (the point the PC targeted when activating the item). The creature is then created from the Guenwhyvar template (the template takes its name from the creature's first name; you can view it under Blueprint ResRef in the Advanced tab). And that's it.

Additional enhancements would include placing a visual spell-like effect at the target location, representing the summoning. As it is now, the summoned creature slowly fades in. Another would be to modify the summoned creature object by setting a local variable (as part of the summoned creature object) to 0 upon creation, using an **OnSpawn** script. You could then write an **OnHeartbeat** script for the object that would increment the counter until a certain value is reached and then destroy the summoned creature (so if you set the value to 10, Guen would last 10 turns). This is useful for keeping the gaming experience balanced for the PC, and accurately representing Guenwhyvar's abilities.

WHAT'S UNDER THE HOOD?

If you do any scrounging or scavenging in the script files for the *NWN* campaign you will see #include statements and other functions not listed in the Function list of the Script Editor. For example, the following script is for the Recall Stone given to you in Chapter 1 of the campaign.

Neither the included file (NW_I0_Plot) or the IsRecall( ) function can be found using the default settings of the Script Editor. Basically, an #include line automatically inserts code, from the specified file, into the current script when the script is compiled. You can use this feature to put additional code (used by many different scripts) into one file and then add that file to any script needing that code. This saves a ton of typing and makes your scripts more readable.

In this example, IsRecall( ) is defined in the NW_I0_Plot file. The func-

SCRIPT CODE

```
#include "NW_I0_Plot"

void main()

{

     IsRecall();

}
```

tion is rather large, so I won't include it here but basically, it sets up a jump for the activator of the stone back to the nearest Temple of Tyr, if one exists.

However, finding this file is rather difficult. This is because the default setting only lists the resources in the module currently loaded; not the entire toolset. In order to examine all the resources, you have to modify the settings a bit using the Script Editor.

In the Script Editor dialog box, click on the Open an Existing Script button. In the Select Resource dialog box, select All Resources, from the set of radio buttons on the lower left-hand side. At this point, you will have a list of all resources in the toolset. To find NW_I0_Plot (or any other include file you are looking for) enter the name in the Resource Name control and click OK. Once this file is open, you can search for IsRecall( ), using the Find text tool in the Script Editor. In addition to the complete listing, you can filter the resources shown by selecting options from the Name Filter control. For instance, to just see the Include files (like NW_I0_Plot), select Include_Files.

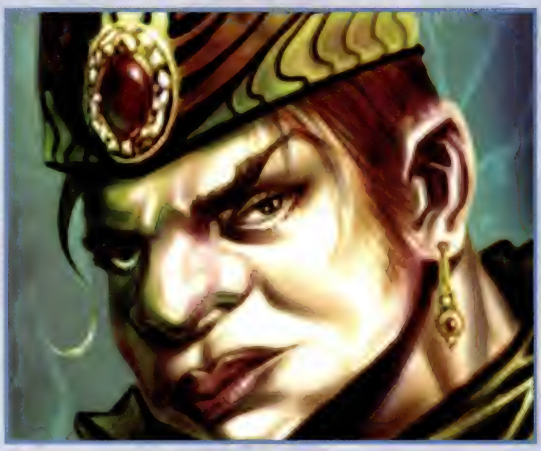
Using this setting allows you to examine all the code BioWare provided in their toolset. This is incredibly powerful, as many questions can only be answered using these settings.

Scripting The Entering and Exiting Of Clients

In order to adventure in a campaign, you must first log onto a *NWN* server and join a new or ongoing game. When you (the client) enter or exit the game, two scripts are triggered for these events.

The **OnClientEnter** handler is called when any client enters the module. This provides the module designer with a chance to perform any actions before the PC appears in the module. Several possibilities include:

- Determining the identity or any characteristics of the entering PC, such as class, race, or name. These could then trigger events in the game such as a special encounter. (Biff [Fighter, level 8] enters the game, triggering a meeting with an NPC who has a special grudge with Biff from a previous campaign.)
- Determining the level of the entering PC and rewarding (or punishing) them in such a way as to even the odds a little. You can't alter the PC's stats, but you could give a low-level PC a powerful item, for example.
- Marking the PC in some way that triggers a special conversation with a pivotal NPC.
- Adding (or removing) a special item, available only within the module.



Example OnClientEnter Script

The following script is taken from Chapter 1, and is a simple script that "cleans up" the PC. You'll see what I mean after reading through it:

```
void main()
{
  //Declare major variables
  object oTarget = GetEnteringObject();
  effect eVisual = EffectVisualEffect(VFX_IMP_RESTORATION_GREATER);

  effect eBad = GetFirstEffect(oTarget);
  //Search for negative effects
  while(GetIsEffectValid(eBad))
  {
```

```
    if (GetEffectType(eBad) == EFFECT_TYPE_ABILITY_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_AC_DECREASE ||
      /*  other effect type checks deleted for brevity  */
      GetEffectType(eBad) == EFFECT_TYPE_NEGATIVELEVEL)
    {
      //Remove effect if it is negative.
      RemoveEffect(oTarget, eBad);
    }
    eBad = GetNextEffect(oTarget);
  }
  if(GetRacialType(oTarget) != RACIAL_TYPE_UNDEAD)
  {
    //Apply the VFX impact and effects
    int nHeal = GetMaxHitPoints(oTarget) - GetCurrentHitPoints(oTarget);
    effect eHeal = EffectHeal(nHeal);
    ApplyEffectToObject(DURATION_TYPE_INSTANT, eHeal, oTarget);
  }
  //Fire cast spell at event for the specified target
  ApplyEffectToObject(DURATION_TYPE_INSTANT, eVisual, oTarget);
}
```

This code basically grabs the entering object (using **GetEnteringObject( )**) and stores a visual spell effect (Greater Restoration) in the **eVisual** variable. Then a **while** loop is used to find all bad effects (such as ability or level drain) and fix them one at a time. Once all bad effects are "cured" (using a call to **RemoveEffect( )**), the **while** loop exits.

Next, the object is checked for the Undead racial type. If the object is not Undead (**if(GetRacialType(oTarget) != RACIAL_TYPE_UNDEAD)**) then the object is completely healed.

Finally, to indicate that all this crazy cleaning and healing occurred, the Greater Restoration visual spell effect is applied to the object, using the **ApplyEffectToObject ( )** function. The end result is that the entering object assumes that Greater Restoration, a 7th level Cleric spell, was just cast on them.

The **OnClientExit** hook is the exact opposite. It is called whenever a client exits the module. The possibilities for this script are the same as **OnClientEnter**.

Scripting OnHeartbeat For Modules

The **OnHeartbeat** event handler is called every 6 seconds (one turn in game time). For individual areas (which also handle this event), this is a very useful hook that we will explore later. But for module-wide effects, there aren't too many uses for an **OnHeartbeat** script.

The only times you would use a Module OnHeartbeat event, as opposed to an Area OnHeartbeat event, would be when you want an effect to apply to a large group of areas. For instance, a large town or city would have several areas close together, so a random city effect (such as random robberies or muggings or the initiation of patrols for city guards) would be a good idea for an **OnHeartbeat** script.

Example OnHeartbeat Script

The following script was taken from Chapter 4 of the NWN campaign, and illustrates the possibilities:

```
void main()
{
  if (GetLocalInt(GetModule(), "NW_G_M4Q01_CATAPULTS") <= 2)
  {

    object oTarget = GetNearestObjectByTag("M4ATarget", GetObjectByTag("M4ASource"),
Random(6)+1);
    if (GetIsObjectValid(oTarget) == FALSE)
    {
      oTarget = GetObjectByTag("M4ATarget");
    }
    AssignCommand(GetObjectByTag("M4ASource"), ActionCastSpellAtObject(SPELL_FIREBALL,
oTarget,METAMAGIC_ANY, TRUE));
  }
}
```

This script is used to simulate a city under siege by catapults. In this case, a module variable (**NW_G_M4Q01_CATAPULTS**) is used to track the related quest and ensure the player has not yet destroyed all the catapults. If the quest is still active, a catapult target is selected randomly from the six available. These targets are six waypoints scattered throughout the area map. The source is an invisible object (a placeable item type), cleverly placed atop the central tower of the area map. The target is checked for validity and if not valid, a "backup" target is selected. At this point, it's time to rain fiery death down upon the city. Using the **AssignCommand( )** function, the source (the object atop the tower) casts a Fireball spell at the randomly selected target. As this occurs every six seconds in a fairly random location, it provides excellent flavor to the city-under-siege environment.

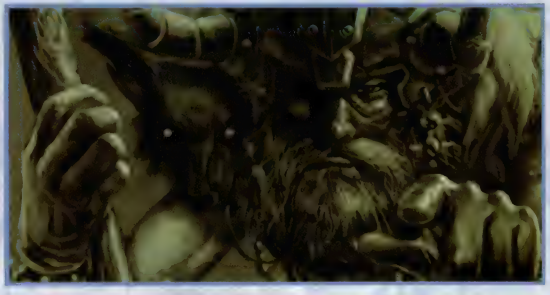
Scripting OnModuleLoad For Modules

The **OnModuleLoad** event handler is called when the module is first loaded. As mentioned previously, the script can set up the initial state (such as local variable values) of the module and/or included areas.

Scripting For Player Characters

Probably the most important of the event handlers, the following are vital to scripting for PCs:
OnPlayerDeath – called when the PC has –10 HP, or lower.
OnPlayerDying – called when the PC is unconscious (between 0 and –10 HP).
OnPlayerRespawn – called when the PC chooses the Respawn option from the *NWN* game panel.

These hooks determine the behavior of the PC at those critical moments when they take a +4 long sword to the back of the head, trigger a deadly poison gas trap, or meet an Ancient Blue dragon.



Example Dying, Death, & Respawn Scripts

The following scripts are taken from the *NWN* campaign, and are considered "standard behavior."

OnPlayerDying Script

```
void main()
{
  effect eDamage = EffectDamage(11, DAMAGE_TYPE_MAGICAL, DAMAGE_POWER_PLUS_FIVE);
  ApplyEffectToObject(DURATION_TYPE_INSTANT, eDamage, GetLastPlayerDying());
}
```

This script is pretty ruthless, as it automatically kills the PC if the PC falls below 0 HP. Using **ApplyEffectToObject( )**, 11 damage (of type magical) is delivered to the last PC found dying. No need for a healer NPC in this module! Note that the **DAMAGE_POWER_PLUS_FIVE** parameter negates any Damage Reduction of the PC object.

OnPlayerDeath Script

```
void main( )
{
  // * make friendly to each of the 3 common factions
  object oPlayer = GetLastPlayerDied( );
  AssignCommand(oPlayer, ClearAllActions( ));

  if (GetStandardFactionReputation(STANDARD_FACTION_COMMONER, oPlayer) <= 10)
  {  SetLocalInt(oPlayer, "NW_G_Playerhasbeenbad", 10); // * Player bad
    SetStandardFactionReputation(STANDARD_FACTION_COMMONER, 80, oPlayer);
  }

/* the same action is taken with the Merchant and Defender factions */
  PopUpGUIPanel(GetLastPlayerDied(),GUI_PANEL_PLAYER_DEATH);

}
```

The object for the last PC to die is retrieved using **GetLastPlayerDied( )** and then all remaining actions are cleared from the stack. This prevents the PC from doing wacky things upon being raised or resurrected immediately (or in another area). The reputation of the PC is then reset, if less than 10, for the Commoner, Merchant, and Defender factions. Finally, the Player Death menu is displayed. The PC then chooses from several choices; one of them being respawn. What a nice segue to our next script hook!

Note

In the interest of clarity, I included only the ApplyPenalty( ) function from the OnRespawn script. For a complete listing of this script, see nw_o0_respawn in the official campaign's Chapter 1 module.

OnPlayerRespawn Script

```
void ApplyPenalty(object oDead)

{

  int nXP = GetXP(oDead);

  int nPenalty = 50 * GetHitDice(oDead);

  int nHD = GetHitDice(oDead);

  // * You can not lose a level with this respawning

  int nMin = ((nHD * (nHD - 1)) / 2) * 1000;


  int nNewXP = nXP - nPenalty;

  if (nNewXP < nMin)

    nNewXP = nMin;

  SetXP(oDead, nNewXP);

  AssignCommand(oDead, TakeGoldFromCreature(FloatToInt(0.10 * GetGold(oDead)), oDead,

TRUE));

}
```

This function is interesting because it penalizes the PC if it chooses to respawn. It is automatically called (in addition to that "cleaning" function we discussed in the **OnClientEnter** script) by the main portion of the **OnRespawn** script.

The function works as follows: the current XP of the dead player is retrieved. The penalty is calculated, based on the HD of the PC and a complex formula guaranteeing no level loss for the PC. The XP of the PC is then set to the lower of two values: **nNewXP** or **nMin**. Finally, 10% of the gold carried by the PC is taken (kind of like the federal government) as another penalty.

The script continues by cleaning up the PC and then respawning it at various locations (based on special circumstances), or at the point the PC last died.

On Death and Dying

The issue of death and dying is one that every module designer must address, and since it will have a huge impact on the design of your adventure, the sooner you make the decision the better. Unless your module features a cameo from good old Sergol, the wizard who waits patiently in Neverwinter's Hall of Justice for players to die, you'll need to come up with a new system of respawning fallen PCs. Here are a few options we came up with (and one we implemented):

The Instant Respawn System: Immediately respawn at the point you die. This is easy to set up, but definitely not recommended. It leads to situations where players who are getting their butts kicked in an impossible fight respawn, resume getting their butts kicked, die again, and respawn over and over again, all the while accumulating huge XP penalties and massive amounts of resentment toward the module designer. If you remove the penalties, this system is equally terrible, because players are no longer penalized for bad play, and then what's the point?

The Single Respawn Point System: Your module has one place where players respawn; usually a temple or the starting point of the module (it was the Hall of Justice in Chapter 1 of the official campaign). To implement this, use the **OnDeath** and **OnDying** scripts from the official campaign and a simple **OnRespawn** script like the one shown as **Figure 1.** You can also put an NPC or object at the respawn point, offering to send you back to the point where you died, as Sergol did in the official campaign. We'll look into that at **Figure 4.**

The Save Point System: As in many console games (think of the glowing save points in the *Final Fantasy* series), you can set up a series of objects that change a local variable when their **OnUse** script is fired (or some other event common to all objects). This local variable must be initialized before the PC enters the module for the first time (usually in the **OnClientEnter** event handler for the module). This allows players to determine what their respawn point will be, and allows the module designer to modulate game difficulty by changing the distance between save points. We cover this system in **Figures 2 and 3.**

The "Just Screw 'Em" System: Hey, if your NPC's didn't *want* to be dead, they shouldn't have pick-pocketed that Balor Lord! If players are playing alone, they'll be forced to load a saved game. If they're playing multiplayer, they can wait for their friends to re-animate them. To implement the "screw 'em" method, just copy the default **OnDeath** and **OnDying** scripts and leave the **OnRespawn** handler alone.

Example Scripts

Figure 1

```
void RemoveEffects(object oDead)

{

  //Declare major variables

  object oTarget = oDead;

  effect eVisual = EffectVisualEffect(VFX_IMP_RESTORATION);

  int bValid;


  effect eBad = GetFirstEffect(oTarget);

  //Search for negative effects

  while(GetIsEffectValid(eBad))

  {

    if (GetEffectType(eBad) == EFFECT_TYPE_ABILITY_DECREASE ||
```

```
      GetEffectType(eBad) == EFFECT_TYPE_AC_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_ATTACK_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_DAMAGE_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_DAMAGE_IMMUNITY_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_SAVING_THROW_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_SPELL_RESISTANCE_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_SKILL_DECREASE ||
      GetEffectType(eBad) == EFFECT_TYPE_BLINDNESS ||
      GetEffectType(eBad) == EFFECT_TYPE_DEAF ||
      GetEffectType(eBad) == EFFECT_TYPE_PARALYZE ||
      GetEffectType(eBad) == EFFECT_TYPE_NEGATIVELEVEL)
      {
        //Remove effect if it is negative.
        RemoveEffect(oTarget, eBad);
      }
    eBad = GetNextEffect(oTarget);
  }
  //Fire cast spell at event for the specified target
  SignalEvent(oTarget, EventSpellCastAt(OBJECT_SELF, SPELL_RESTORATION, FALSE));

  ApplyEffectToObject(DURATION_TYPE_INSTANT, eVisual, oTarget);
}

void main()
{
  object oRespawner = GetLastRespawnButtonPresser();
  ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectResurrection(),oRespawner);
  ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectHeal(GetMaxHitPoints(oRespawner)),
oRespawner);
  RemoveEffects(oRespawner);

  location respawnLoc;
```

```
// Return PC to specific WP

   respawnLoc= GetLocation(GetObjectByTag("cf_waypt1"));

   AssignCommand(oRespawner,JumpToLocation(respawnLoc));

}
```

This script contains two functions: **main( )** and **RemoveEffects(object oDead)**. The **main( )** function first fully resurrects and heals the PC. Then the **RemoveEffects( )** function is called, removing any negative effects the PC was experiencing when he or she died. Before returning, **RemoveEffects( )** casts a "fake" Greater Restoration spell on the PC. After the PC is "cleaned up," the respawn location is retrieved (in this case, **cf_waypt1**) and used in a call to **AssignCommand( )**. This first parameter is the newly-recovered PC object. The second parameter is the command being assigned to the PC; in this case, **JumpToLocation( )** (with the special waypoint location).

Figure 2

```
void main()

{

   SetLocalString(GetModule(), "RespawnPoint",GetTag(OBJECT_SELF));

}
```

This script is very small and is placed in the **OnUse** event handler of the object representing the savepoint item. It updates a local variable (**RespawnPoint**) in the module object with the tag of the object just touched by the PC. As noted before, you must initialize this local variable before the PC enters the module. You then place copies of this object at key points in the game, saving the player's valuable progress.

The following script (placed in the **OnPlayerRespawn** event handler of the module) is very similar to the version of the **OnPlayerRespawn** script used in the **Single Respawn Point System**, with one small difference. The respawn location is updated by retrieving the current value of **RespawnPoint** and then used by **JumpToLocation( )** to return the player to the last save point. The code for **RemoveEffects( )** is not included in this sample in order to focus on the modified code of the **main( )** function.

Figure 3

```
void main()

{

   object oRespawner = GetLastRespawnButtonPresser();

   ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectResurrection(),oRespawner);

   ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectHeal(GetMaxHitPoints(oRespawner)),
oRespawner);

   RemoveEffects(oRespawner);


   location respawnLoc;


   // Return PC to last savepoint

   string curRP;
```

```
    curRP= GetLocalString(GetModule(), "RespawnPoint");


    respawnLoc= GetLocation(GetObjectByTag(curRP));

    AssignCommand(oRespawner,JumpToLocation(respawnLoc));

}
```

The following is the script that is called by Sergol's conversation file if you die in Chapter 1. You can check it out in area M1Q1F, conversation file "m1q1fdeath," script nw_d1_deatht_3. So this is how a "Word of Recall" spell really works! Feel free to copy and paste it into an appropriate Sergol character or item in your module.

Figure 4

```
// * transport player back to place where you died

void main()

{

    object oSelf = OBJECT_SELF;

    effect eVis = EffectVisualEffect(VFX_IMP_UNSUMMON);

    ApplyEffectAtLocation(DURATION_TYPE_PERMANENT, eVis,

GetLocalLocation(GetPCSpeaker(),"NW_L_I_DIED_HERE"));

    SetLocalInt(GetPCSpeaker(), "NW_L_I_DIED", 0);

    object oPC = GetPCSpeaker();

    ActionCastFakeSpellAtObject(SPELL_FREEDOM_OF_MOVEMENT, OBJECT_SELF);


    AssignCommand(GetPCSpeaker(),  DelayCommand(0.5,

JumpToLocation(GetLocalLocation(oPC,"NW_L_I_DIED_HERE"))));

}
```

Death in 3rd Edition

To explain the difference between Death and Dying, we'll need a refresher on 3rd Edition rules. When you have between -1 and -9 hit points, your character is dying. A dying character falls unconscious and can take no actions. A dying character continues to lose 1 hit point a round until he stabilizes (stops losing hit points), or die. Every round, you have a 10% chance of stabilizing. If someone successfully heals you (through a spell or the Heal skill), you stabilize automatically. If that healing puts you at above 0 hit points, you're alive again.

If you ever have -10 or fewer hit points, you're dead. No one can heal you, because your spirit has left your body. The only recourse is a high-level spell like Raise Dead or Resurrection.

You may have noticed that it doesn't work this way in the official campaign. Whenever you're dying, the default **OnDying** script activates a function that automatically kills you, sparing you the tedium of watching your character slowly bleed to death. But if you're a stickler for *D&D* rules and you know that there will be other players capable of helping a dying character, you can go ahead and delete the default **OnDying** script. The 3rd Edition rules will then work as intended. But for the purposes of our examples, we'll be using BioWare's lovely kill-'em-when-they're-down **OnDying** script, and the clean-up-the-reputation-and-bring-up-the-GUI **OnDeath** script.

Scripting User-Defined Behavior for Modules

In the previous sections, we have examined a large variety of script hooks for various events that occur in the game. Some scripts are called frequently (every 6 seconds for the **OnHeartbeat** script and every time an object is picked up or dropped by a PC) and others are (hopefully) called infrequently, such as the **OnPlayerDeath** and **OnPlayerDying** scripts. The last event handler (for modules) we will examine is perhaps the most interesting; the **OnUserDefined** event handler. Similar to the **OnHeartbeat** script, the **OnUserDefined** script is available to every object in the game, as well as the module and areas.

Unlike the others, this script was designed solely for usage by the campaign developer; in other words, you and I. The handler is designed to handle singular, custom events (or anything you can imagine) in your campaign. It is such a convenient hook because it is accessible from **anywhere** in the game. Once you get the object that owns the user-defined script, you can party on it to your heart's content! But enough talk. Let's check it out for ourselves.

In terms of the module, the user-defined script usually contains significant events that aren't tied to any single area. If they were, they would be in the *area's* **OnUserDefined** script hook.

Before digging into a real sample, let's go over a few basics regarding user-defined scripts and their related events.

Each user-defined script is usually a large **switch** statement, with each case being a different custom event. **Switch** statements are used because they are very easy to expand. Handling additional custom events only requires a new **case** statement, and code for the custom event. Expanding an **if-then-else** statement is harder and could cause more harm than good. It's always good to plan ahead when designing your campaign. Invariably, you will go back to some "finished" area and add new features.

Once a user-defined script is created, you execute the custom events by making calls to **SignalEvent( )** and **EventUserDefined( ),** from other scripts in the campaign. For instance, let's say that you want to have a monster come running when a certain treasure chest is open (damn dirty thieves!). This is accomplished by writing a script for the **OnOpen** script hook of the chest. Inside this script, you would make a call to **SignalEvent( )**. This function, provided by BioWare, has two parameters: the object receiving the custom event and the identity of the custom event (let's say 1001). The second parameter is a little more complex because it's actually a call to **EventUserDefined( )**. **EventUserDefined( )** informs the object receiving the event that the current event is a custom event and should be handled by the object's **OnUserDefined** script. The parameter for **EventUserDefined( )** is, in our fictitious example, the identity of the event (1001). Basically, the entire concept involves two objects (one signaling the custom event and one receiving the custom event) and two scripts (one script sending the event using **SignalEvent( )** and the **OnUserDefined** script of the object receiving this event).

Let's nail the concept down with two scripts taken from the NWN campaign.

Example OnUserDefined Script

First, let's examine the user-defined script for the Chapter 1 module.

```
void main( )
{
  int nEvent = GetUserDefinedEventNumber();
  int nNth;
  switch(nEvent)
  {
    case 600:
      DestroyObject(GetObjectByTag("m1Q04CHelmite"));
      DestroyObject(GetObjectByTag("M1Q02Chelmite"));
      DestroyObject(GetObjectByTag("M1Q05Helmite"));
      DestroyObject(GetObjectByTag("M1Gen_HumPriest"));
```

```
        nNth = 0;
        object oHelmiteEncounter = GetObjectByTag("ENC_M1Helm",nNth);
        while(GetIsObjectValid(oHelmiteEncounter))
        {
          SetEncounterActive(TRUE,oHelmiteEncounter);
          nNth++;
          oHelmiteEncounter = GetObjectByTag("ENC_M1Helm",nNth);
        }
      break;
    }
}
```

In the sample above, a simple encounter is set up and activated, due to some other event in the *NWN* campaign (which is the next sample). The script can be broken down as follows

- The custom event number (or its ID) is retrieved using GetUserDefinedEventNumber( ). For our example, let's say the ID is 600.
- The switch statement is then entered and a match is found for the event number 600.
- The matching case statement is executed. Several Helmite priests are destroyed and then several Helmite encounters are retrieved and activated.

This series of steps covers the basic format for executing all user-defined scripts, or at least those using a **switch** statement. Now let's examine the other side of this; the cause of the custom event.

In order to receive the request to run the custom event, identified by 600, there had to be a script that created the request and fired it off.

The Executing Script

```
void main( )
{
  //Summon Mullond
  object oAttendant = GetLocalObject(GetModule(),"NW_G_M1Q5Atte");
  SignalEvent(oAttendant, EventUserDefined(110));

  //Open mystery building
  object oDoor = GetObjectByTag("M1Q1_M1Q6A1");
  AssignCommand(oDoor,ActionOpenDoor(oDoor));

  // Leave
  SignalEvent(OBJECT_SELF, EventUserDefined(111));

  //Update Plot
  SetLocalInt(GetModule(),"NW_G_M1Q1MainPlot", 100);

  // send our mystery custom event
  SignalEvent(GetModule(),EventUserDefined(600));
}
```

One of the first things you should have noticed is that several custom events are fired, in addition to our custom event 600. Obviously, this script is attached to a fairly important in-game object. However, we are only concerned with the last two lines of code.

As I said before, a custom event is sent using a call to **SignalEvent( )**. In this sample, the first parameter (the object to receive the event) is the Chapter 1 module of the *NWN* campaign. Therefore, we retrieve the object representing it with a call to **GetModule( )**.

NOTE

When we re-examine custom events for objects, this call is usually replaced by a call to GetObjectByTag( ).

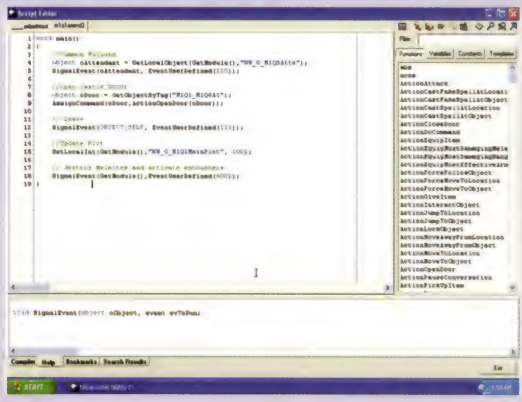
The second parameter is the event number; in this case 600. Once these parameters are set up, the script fires off the custom event to the module's **OnUserDefined** script.

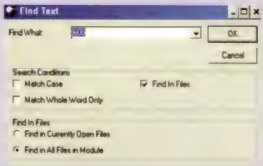
Finding Text in the Script Editor

An incredibly useful feature of the Script Editor, the Find Text feature, was used to retrieve this exact script from the game. To access this, open up the Script Editor from the Tools menu. Once the Script Editor is open, notice the small toolbar on the right side.

Click the first magnifying glass button to open the Find Text dialog box.

This dialog is incredibly useful for finding any text in any script in the current module. The coolness kicks in when you are trying to figure out how a certain function or predefined constant is used in the game. Enter the item to search for, checking the Find in Files checkbox. The dialog box expands, with two radio buttons. We are mostly interested in the second, Find in All Files in Module, so go ahead and select it. Now click OK and you should eventually get a list of scripts, and the line containing your search text. Double-click on any one of the results and the script is opened, allowing you to examine the entire script! This becomes incredibly powerful when digging around in the BioWare modules (and any other large module). This was, **by far**, the most useful tool we found in terms of scripting.





Using the Find Text tool, you could continue your research of this script by finding all scripts that receive the custom event 110, by searching for "case 110". Make sure that you select Find in All Files in Module or you will only search in the present area file that is open.

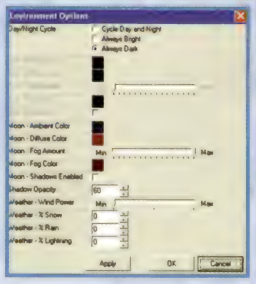
That about covers it for module-level scripting. Let's move on to an explanation of the Area Properties, followed by the smaller set of Area scripts.

3 Area Properties

Yawn... And here I bet you thought we'd start our module creation process off with something exciting and fun to pull you in. Instead, you got twenty pages of Module Properties and Scripting. Fortunately, as graduates of our Basic Module Building chapter will know, making Areas is a lot more fun. Let's get back to that before our brains shut down entirely!

Visual Area Properties

If you want to modify a default, you can hit the Customize Environment button to bring up the Environment Options window. Let's take a deeper look at the options available here.



Day/Night Cycle: Note that choosing Always Bright or Always Dark won't freeze time, it simply locks the area into a consistent lighting scheme. Make sure to choose Always Dark for underground areas that have no natural light.

Sun/Moon Ambient Color: The color of reflected light in a room. This often mimics the main colors in an area, like brown and green in a forest. Dark areas have very little ambient light, so use a dark color or black. Ambient colors can also be used to simulate unusual light sources like lava or radioactive waste.

Sun/Moon Diffuse Color: The color of the main light source which strikes the floor and tops of objects. Outdoors, this is the sun or moon, indoors it would need to be some ceiling mounted light or sunlight from high windows.

Sun/Moon Fog Amount: Fog can make navigation a pain for players, so don't overdo it unless you really want the effect.

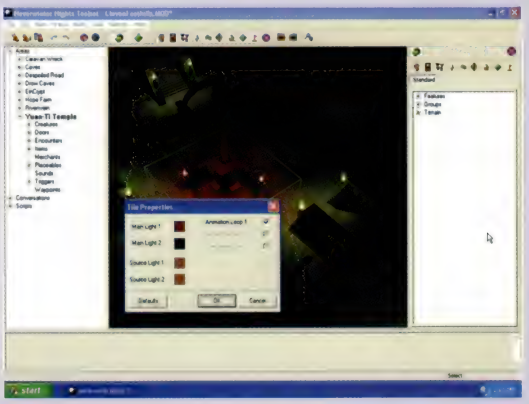
Sun/Moon Fog Color: Fog doesn't seem to catch light sources, so you'll need to tweak this to match your scene, or else you'll have white mist in a dark forest. You can also make the fog seem like something else—the blue/grey particulate created by a series of waterfalls on a sunny day, a suspicious purple cloud of poison gas, or the thick black mist of an area blanketed by a Darkness spell.

Sun/Moon Shadows: In a bright area with lots of reflected light, shadows are dim or barely visible. The moon's light is rarely bright enough to cast shadows. Turn 'em on or off here.

Shadow Opacity: This is an interesting one; in a dark area with exactly one light source, the Opacity should be 100%. In a bright area saturated with reflected light, it would be 10% or less. This is a great way of underscoring the lack of reflected light in a dark cave or crypt.

Weather: This should all be pretty self-explanatory: You just set the odds that a player will experience each weather effect when they enter the area.

Lighting Your Areas



We didn't really cover this much in the Basic section, so we'll devote a little time to it here. Lighting is, in my opinion, the key to making distinctive areas. Everyone who puts a crypt in their module will have access to the same crypt tileset, but if I use a creepy green glow in a fogless area and you use a few harsh sulfuric lights in an area of thick black mist, our areas will feel drastically different. I encourage everyone to take the time to experiment with both the **Customize Environment** options and the individual tile lighting options to give their areas new layers of impact.

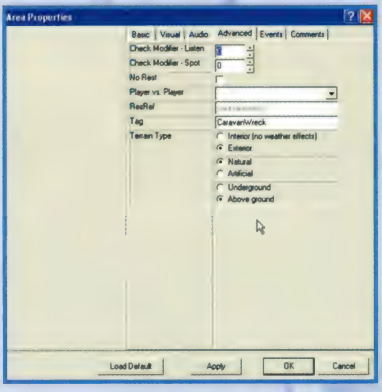
Remember that each tileset has its own light sources, which can be accessed by right-clicking on the placed tile. Every area has two Main Lights: **Main Light 1** pervades the tile, as if it were a bright bulb in the center, casting its light in a sphere that dissipates as it reaches the edge of the tile. **Main Light 2** is a spotlight overhead, casting its light directly down on the tile. It usually dissipates before it hits the floor, so only objects that protrude a little higher will be touched by its light.

To turn either Main Light off, choose black as the color (most tiles have this as their defaults). When you turn them on, think of the real-life source that would be casting that light; outdoor ruins on a night with a full moon but no other light source would have a white Main Light 2 but Main Light 1 would be dark or black. An underground crypt like our Yuan-Ti temple would have no Main Light 2, but the unnerving colors of the ceremonial torches would pervade the temple as the Main Light 1 in certain tiles.

Some tiles have **Source Lights** as well. These are light sources that are actually depicted in the tile: torches on the walls, braziers in the corners, lamps on the tables, etc. You can turn these off and edit their colors in the same way. Many tiles have **Animation Loops**, like branches swaying in the wind, that can also be turned off.

Remember that your scene will then be tinted by the Ambient, Diffuse, and Fog Colors set in the **Environment Options**, and further illuminated by whatever light-sources the player carries with them. With six different factors determining the color balance of each tile, you have a stunning array of possibilities.

Advanced Area Properties



Check Modifiers: Even the sharpest-eyed character can't see much in dense fog. And that crashing waterfall may drown out the sounds of other beings in the area. Set your Listen and Spot penalties here. For example, jack 'em up for a rural area with no obstacles on a weatherless day when sound carries forever.

No Rest: A very important option. By setting areas to No Rest you force players to play carefully, conserve spells, and rely on potions and each other for healing. If you're going to use this option, use it consistently. For instance, players can rest above ground but not in caves, crypts, and sewers. Otherwise, your players will be baffled and annoyed (as opposed to merely being annoyed).

Player vs. Player: Another major issue. This should be consistent throughout the campaign, unless there's a consistent rule to explain otherwise (no fighting in towns, but anywhere else is okay; no fighting except in designated dueling grounds, etc.).

- No PvP – Players cannot damage other players.
- Party Protected – Players cannot damage players in their party, but can attack other parties.
- Full PvP – The gloves are off! Keep your eyes on the rogue.
- Default – Defer the decision to the host of the game.

Terrain Type: If you did something strange, like using the crypt tileset to make what is supposed to be an outdoor area, you can change the terrain type here so that certain feats (like the ability of Druids and Rangers to move silently in wilderness areas) work correctly.

4 Area Scripting

Area scripting has a smaller set of event handlers, compared to module scripting, but they are just as useful! All area event handlers are available using the Events tab of the Area Properties dialog box.

Note

You can also open the Area Properties dialog box by right-clicking any area node and selecting Properties from the context menu.

The four handlers for areas reflect the transition characteristic, and others:

OnEnter and **OnExit scripts** – called when the PC enters or exits the area.

OnHeartbeat script – called every 6 seconds (of game time). Useful for providing interesting effects, such as a poison gas room.

OnUserDefined script – Called by the **SignalEvent( )** scripting function. Useful for implementing any custom behavior. This event handler is also available to all other programmable objects in the game.

Scripting for Entering and Exiting an Area

Similar to entering and exiting a module (**OnClientEnter** and **OnClientExit**), the **OnEnter** and **OnExit** event handlers are called when the PC enters or exits an area in the module. Here are some possibilities for enter and exit scripts:

- Modifying the current characteristics of the PC (such as healing or harming them), based on the storyline or events within the area.
- Adding a new quest to the journal of the entering PC, or updating an existing quest when exiting.
- Revealing the entire area map upon entry. For example, when entering a city or town for the first time. This is actually done when the PC enters Neverwinter for the first time in Chapter 1.

Example OnEnter Script

Here are a few examples of an **OnEnter** area script taken from the *NWN* campaign:

```
void main( )
{
   ExploreAreaForPlayer(OBJECT_SELF,GetEnteringObject());
}
```

This is a very simple script, but I thought it would be a good idea to include this script, as the function is pretty handy for modules that take the PC through several towns and/or cities. The following is an example of adding a new quest for the entering PC. This is very useful for getting the ball rolling in the first area of a new module.

```
void main( )
{
   object oPC = GetEnteringObject();

   if (GetIsPC(oPC) && (GetLocalInt(oPC, "NW_JOURNAL_ENTRYJOUR_M1Q0Training") < 10))
   {
      AddJournalQuestEntry("JOUR_M1Q0Training", 10, oPC, TRUE, TRUE);
   }
}
```

This isn't too complex but, once again, shows off some functions you will probably be making quite a few calls to:

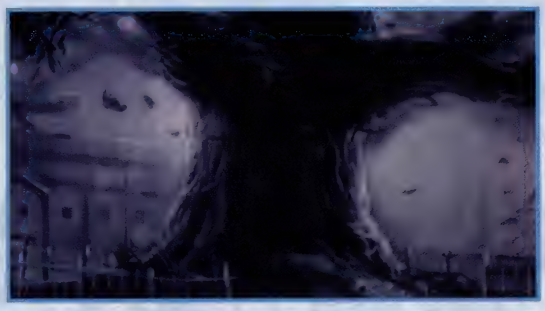
GetEnteringObject( ) – retrieves the most recent entering object of the area.

GetIsPC( ) – just like it says, determines if the specified object is a PC object.

GetLocalInt( ) – this (and its counterpart, **SetLocalInt( )**) get or set local properties on the specified object.**AddJournalQuestEntry( )** – adds a new entry to the journal of the specified PC.

Except for a few minor details, there's not much left to explain about this script. First of all, notice that it is not assumed that the entering object is the PC. It is checked for validity before any access or modification. This is good practice as there are other objects that enter areas (such as NPCs teleporting in, monsters spawning, and the faithful henchman or familiar of a PC). It wouldn't be very helpful to update the journal for the raven familiar of a wizard with a new quest, now would it?

The second detail is the modification of the "Training" quest. Before a modification is made, the journal first checks for the presence of the "Training" quest. If this wasn't done, the "Training" quest entry would be reinitialized every time the PC wandered back into this area (not very good for your reputation as a module designer). If the check is successful (the object is a valid PC and the Training quest is not present), the Training quest is added with a current status of 10 (this is how quest progress is tracked in the journal) to all current player characters. This "party-wide" update is caused by the **TRUE** value of the last two parameters.



Example: Linking Encounter Status to a Quest

At the beginning of our module, the player's caravan is attacked by the Yuan-Ti and their Bugbear thugs. The player must try to rescue the scattered survivors of the caravan, many of which have been taken as slaves by the Yuan-Ti, back through the caves from which they entered. As you'd imagine, there are a number of Yuan-Ti and Bugbear encounters set up in this area. But on play-testing, we found a problem; what if the player came back to this area weeks after the initial attacks, and triggered an encounter he or she avoided the first time? The Yuan-Ti don't attack at random, so it wouldn't make sense for the encounter to fire. We wanted to destroy all the remaining encounters when the "Search for Survivors" part of the quest was complete.

In other words, if the PC has not completed a certain quest, all encounters in the current area are active. When the PC completes the quest, all encounters in that area or areas are deactivated. The solution was an **OnEnter** script that monitors the current state of a local variable (stored with the PC and called **J_ENTRY_YUANTIQUEST**) and, depending on the value, activates or deactivates the encounters in the current area. The solution components are as follows:

- A local variable on the PC object, called **J_ENTRY_YUANTIQUEST**, indicating the current state of the Yuan-Ti quest. This variable must be set or updated before the PC enters the area.
- **cf_cw_actenc** – an area script in the **OnEnter** handler that either activates or deactivates all scripts based on the current value of **J_ENTRY_YUANTIQUEST**.

OnEnter Script

Here is the script in question:

```
//OnEnter script for activating/deactivating encounters in this area

void ActivateEncounters(int bActive);

void main()

{

  object oPC = GetEnteringObject();


  if(GetIsPC(oPC)&& (GetLocalInt(oPC, "J_ENTRY_YUANTIQUEST") < 30))

    ActivateEncounters(1); //activate encounters

  else

    ActivateEncounters(0); //deactivate encounters

}


void ActivateEncounters(int bActive)

{

  object oEnc1, oEnc2,oEnc3,oEnc4;

  if((bActive == 0) || (bActive == 1))

  {
```

```
  object oEnc1 = GetObjectByTag("GrisstBugbears");
  if(GetIsObjectValid(oEnc1))
    SetEncounterActive(bActive,oEnc1);
  object oEnc2 = GetObjectByTag("YuanTiSnipers");
  if(GetIsObjectValid(oEnc2))
    SetEncounterActive(bActive,oEnc2);
  object oEnc3 = GetObjectByTag("YuanTiandBugbear");
  if(GetIsObjectValid(oEnc3))
    SetEncounterActive(bActive,oEnc3);
  object oEnc4 = GetObjectByTag("YuanTiSnipers");
  if(GetIsObjectValid(oEnc4))
    SetEncounterActive(bActive,oEnc4);
  };
}
```

The first thing we do is retrieve the PC object. The script then checks the PC object for validity and checks that the current value of the journal quest variable is less than 30 (not completed). If both these conditions are satisfied, **ActivateEncounters( )** is called, passing 1. If one or both conditions are not satisfied, **ActivateEncounters( )** is called, instead passing 0.

ActivateEncounters( ) then retrieves each encounter in the area and activates it, according to the value passed in from **main( )** function of the script.

This solution can be customized for usage in different areas or one or two areas. In addition, the triggering conditions can be changed from completion of a quest to any other event in your module.

Example OnExit Scripts

Apparently, the **OnExit** script is not one that is often used. After searching with the soon-to-be-famous **Find Text** tool, I found only the following two examples, taken from the *NWN* campaign:

```
void main( )
{
  if(GetIsPC(GetExitingObject()))
  {
    SetPlotFlag(GetExitingObject(),FALSE);
  }
}
```

This script is interesting because of the call to **SetPlotFlag( )**. This function either turns the Plot property of the specified object on or off. If turned on, the object is invulnerable to all damage. Turned off, the object is so much meat to that Ancient Blue dragon we discussed earlier. So by opening that door, the player casts off his or her immortality, without even knowing it.

Plot Objects

Sometimes the Plot property turns up in amusing scripts. Pavel, an NPC in the Prologue, offers to join you after all hell breaks loose. In his conversation with you he makes the claim that unless you allow him to join you he "will never make it out alive!" Unfortunately, the exact opposite is true because before he joins you he has the **Plot** property turned on. However, the script that adds him as a henchman explicitly turns this property **off**. Fate is a fickle mistress…

Example OnExit Script

Another Example:

```
void main( )
{
  if(GetIsPC(GetExitingObject()))
  {
    int nNth;
    object oDoor = GetObjectByTag("M1Q3COuterDoor",nNth);
    while(GetIsObjectValid(oDoor))
    {
      SetLocked(oDoor,FALSE);
      AssignCommand(oDoor,ActionOpenDoor(oDoor));
      nNth++;
      oDoor = GetObjectByTag("M1Q3COuterDoor",nNth);
    }
  }
}
```

This script was taken from an area in Chapter 1 that had several locked doors. It is related to the dryad reagent quest (specifically **M1Q3PlotReagent**, in the **m1module_1** script).

As with all **OnExit** (and, for that matter, **OnEnter**) scripts, the first thing is to get the exiting or entering object and make sure it is the PC object (remember, assuming makes an ass of u, not me). If true, we then grab the nearest door in the area, identified by the **M1Q3CouterDoor** tag. Checking for validity again (assume), we enter a **while** loop that unlocks the door, opens the door, and then gets the next door with another call to **GetObjectByTag ( )**. If the new door retrieved is valid, then the whole unlocking and opening cycle happens again. This continues until all doors (in this area, there were two) matching the tag are unlocked and opened.

Scripting OnHeartbeat for Areas

This is the exact same event handler as discussed in the Module section. It has the same behavior (triggered ever 6 seconds) and purpose as the module; except for one tiny difference: It is a lot more useful for areas! You will often see this used as a "sentry" for

an area; checking for a certain event or condition and then triggering something in response. Other possibilities include:

- Triggering animation loops for interesting effects, such as a wizard in training testing out his new spells or two townspeople in heated conversation
- Repeating any effect every 6 seconds, such as filling a room with a colorless, poison fog that damages the PC every round.
- Checking for the presence of a PC nearby, in order to start a conversation
- Activating an encounter or scene involving the PC

As you can see, the possibilities are easier to imagine and implement. Let's look at a few examples from the *NWN* campaign.

Example OnHeartbeat Script

This first one is a good example of this script type. It uses a local counter (stored in the area object and called **M1Q1ACounter**) to trigger a custom event (identified by 114 and implemented in the **OnUserDefined** script for the **MAP_M1Q1A** area) every 30 seconds.

```
void main( )
{

  int nCounter =GetLocalInt(OBJECT_SELF,"M1Q1ACounter");
  if(nCounter >= 5)
  {
    SetLocalInt(OBJECT_SELF,"M1Q1ACounter",0);
    SignalEvent(OBJECT_SELF,EventUserDefined(114));
  }
  else
    SetLocalInt(OBJECT_SELF,"M1Q1ACounter",nCounter + 1);
}
```

In case you're interested, custom event 114 spawns a city guard at a certain location in the central district of Neverwinter.

Example OnHeartbeat Script

The following script (in its entirety with modifications) is a trigger with a complicated condition to meet before firing. See if you can figure it out and meet me on other side when you're ready to check your answer.

```
//
//I modified the names of some strings (the NPC tags and the waypoint tags)
//so that everything would fit on one line.
//
```

```
void main()
{
   if(GetLocalInt(OBJECT_SELF,"NW_G_M1Q1A01FireOnce") == FALSE &&
    GetLocalInt(GetModule(),"NW_G_M1Q6A01RitualInProgress") &&
    GetDistanceBetween(GetObjectByTag("TagA"),GetObjectByTag("WP_A")) < 2.0 &&
    GetDistanceBetween(GetObjectByTag("TagF"),GetObjectByTag("WP_F")) < 2.0 &&
    GetDistanceBetween(GetObjectByTag("TagD"),GetObjectByTag("WP_D")) < 2.0)
   {
     SignalEvent(OBJECT_SELF,EventUserDefined(600));
     SetLocalInt(OBJECT_SELF,"NW_G_M1Q1A01FireOnce",TRUE);
   }
}
```

Okay, did you get what was going on here? The massive **if** statement is obviously the main clue. It is testing five conditions:

- Comparing the value of a local variable against FALSE. This determines if the ritual (at the end of Chapter 1 in the *NWN* campaign) referenced inside the **if** statement body has already been run.
- Retrieves another local variable, indicating whether the Chapter 1 ritual is in progress (0, if it is not going on; otherwise non-zero).
- Comparing the distance between three NPCs: Aribeth, Fenthick, and Desther (for example, **TagA**) and the waypoint (**WP_A**) related to them, and 2 meters. Basically, are the NPCs at their proper destinations required for the ritual?

If all these comparisons are valid (or TRUE), then the "Antidote Ritual" can begin (another large script found in the **OnUserDefined** script for the area. Because this ritual should only occur once (and not every 6 seconds), another local variable (**NW_G_M1Q1A01FireOnce**) is set to TRUE, indicating that the ritual has started. Notice that if you immediately go back through the code with the local variable set to TRUE (as a little test for yourself), you would immediately bail out, never performing the ritual.

Scripting User-Defined Behavior for Areas

As with the **OnHeartbeat** script hook, this is an exact match for the module version. Because this is only called when you decide (by using **SignalEvent( )**), you can craft specific effects for an area that trigger many times or only once, depending on your condition. Because it represents the main hook into custom scripting for areas, you can add anything you desire.
Possible ideas for **OnUserDefined** include:

- Updating current quests in the Journal
- Unlocking a series of doors when certain monsters in an area are destroyed
- Activating an encounter when a certain point in a conversation is reached.

Example OnUserDefined Script

Here are a few examples from the *NWN* campaign.

```
void KillAndReplace(object oMark); //this is fully examined in Appendix C.
//
//defined later in the actual script, this activates a series of encounters in the area
```

```
void ActivateEncounters(string sTag);
void main( )
{
  switch(GetUserDefinedEventNumber())
  {
    case 200:
    {
      int nNth = 0;
      object oStudent = GetObjectByTag("M1Q0AStudent",nNth);
      while (GetIsObjectValid(oStudent))
      {
        KillAndReplace(oStudent);
        nNth++;
        oStudent = GetObjectByTag("M1Q0AStudent",nNth);
      }
      KillAndReplace(GetObjectByTag("M0Q0_KETTA"));
      KillAndReplace(GetObjectByTag("M1Q0BElynwyd"));
      KillAndReplace(GetObjectByTag("M1Q0BDendy"));
      KillAndReplace(GetObjectByTag("M1Q0BHewwet"));
      KillAndReplace(GetObjectByTag("M0Q0_HERBAN"));
      KillAndReplace(GetObjectByTag("M1Q0BJaroo"));
      KillAndReplace(GetObjectByTag("M0Q0_BERNA"));
      KillAndReplace(GetObjectByTag("M0Q0_OLGERD"));
      KillAndReplace(GetObjectByTag("M0q01a05Ferd"));
      KillAndReplace(GetObjectByTag("M1Q0ABim"));
      DestroyObject(GetObjectByTag("M1Q0APavel"));
      CreateObject(OBJECT_TYPE_CREATURE,
            "M0Q0_SUMMON_MAGE",
            GetLocation(GetObjectByTag("WP_M1Q0BMage")));
      ActivateEncounters("ENC_M1Q0D17Goblin");
    }
    break;
  }
}
```

This is a great example of an area script because it sets up a surprise for the return of the PC. It is triggered from the graduation room with Aribeth at a certain point in the conversation (by calling **SignalEvent( )** with a value of 200 for the event number), so it is unseen by the PC until the PC re-enters this area.

> **NOTE**
>
> **For the actual code for KillAndReplace( ) and ActivateEncounters( ), see the actual script this was taken from, area_m1Q0b_4, in the Prologue chapter.**

As with all user-defined scripts, this is set up as a **switch** statement. In this case, there is only one **case** statement (affectionately called "The Bloodbath" by Casey) but it's a whopper. The first thing that occurs is that a group of students is killed and replaced by their own corpses. This is a very nice function because it creates individual, varied corpses, instead of the six placeable objects (of type corpse) given to you by *NWN*. Once the students are taken care of, all key NPCs in the Prologue in the area are then killed and replaced. Finally, an evil mage is created and a goblin encounter activated, welcoming the PC back (kind of like that pivotal scene, in a certain space opera that occurred in a place far, far away).

Example OnUserDefined Script

The next script is in it's entirety and demonstrates an ambush by mages and monsters who teleport into a room and attack the PC and company. This is the "certain point in the conversation" mentioned earlier in the discussion of the **KillAndReplace** script:

```
void main( )
{
  switch (GetUserDefinedEventNumber())
  {
    case 100:
    {
      object oGraduate;
      location lLoc;
      int nIdx;
      for (nIdx = 1; nIdx <= 4; nIdx++)
      {
        lLoc = GetLocation(GetObjectByTag("WP_M1Q0CAmbush_0" + IntToString(nIdx)));
        CreateObject(OBJECT_TYPE_CREATURE,"M0Q0_EVILMAGE_02",lLoc);
        ApplyEffectAtLocation(DURATION_TYPE_INSTANT,
                    EffectVisualEffect(VFX_FNF_SUMMON_MONSTER_2),
                    lLoc);
      }
      nIdx = 0;
      oGraduate = GetObjectByTag("M1Q0C_Graduate",nIdx);
      while (GetIsObjectValid(oGraduate))
```

```
        {
          SetPlotFlag(oGraduate,FALSE);
          nIdx++;
          oGraduate = GetObjectByTag("M1Q0C_Graduate",nIdx);
        }
      }
      break;
    }
}
```

This custom event is identified by 100, and is triggered during an important conversation with Aribeth after the PC has graduated. A **for** loop is used to populate four waypoints in the area with an evil mage (there seem to be a lot of those around lately). The waypoint is retrieved with a call to **GetLocation( )** and the evil mage is created with a call to **CreateObject( )**. What's interesting about this is the last call in the **for** loop, **ApplyEffectAtLocation( )**. What this does is "cover up" the pop in of the created mage, making it seem like he (or she, since all evil mages are created equal) teleported in. This is a very common technique used in the scripts of the *NWN* campaign.

After all four evil mages have dropped in, the graduates in the room (who were invulnerable) now lose their special status, by turning the Plot flag off for all of them using another **for** loop (handy little things aren't they?).

And then the classic battle ensues. Which, of course, means that only you and Aribeth are left standing (once again proving that famous old adage: a college degree is worthless once the evil mages start teleporting in!).

Example: Ashes to Ashes (Or Why Greed is Bad in the Underdark)

While exploring the Yuan-Ti temple, players in our module may stumble across a drow infiltrator surrounded by fallen comrades. He eyes the party warily, but is willing to talk, launching an optional sidequest. He claims to be the last surviving member of a scouting party that was obliterated by the Yuan-Ti. The leader of his party, taken by the Yuan-Ti for interrogation, was wearing an amulet that has "sentimental value," and he'd like to get it back. Perhaps you could help? The amulet is merely ornamental, he explains, but he'd give you an enchanted weapon in return. Not a bad deal!

Of course, the Drow is lying, as they so often do. The amulet is an item of great power, as any PC with a high enough Lore score will soon learn. Should you keep it, or return it for a lesser prize? The player is free to do as he or she chooses, but if you choose to pocket the amulet, you may not have it for long. It was made in the underdark, and like so many drow items, will crumble to dust if exposed to direct sunlight.

Clearly, we need a script that destroys the amulet during the day and in exterior areas of the module. If the PC is underground, in an interior area, or it is currently dark, the amulet is safe. The solution is a simple **OnHeartbeat** script for every exterior area in the module. It checks for the needed conditions (daylight, exterior) and then destroys the amulet if found on the PC.

OnHeartbeat Script

The script is as follows:

```
//OnHrtbt for Drow items

void DestroyDrowItems(object oPC);

void main()
```

```
{
    object oPC= OBJECT_INVALID;
    int bIsPCInArea= 0;
    int validTime= 0;

    oPC= GetFirstPC();
    bIsPCInArea= OBJECT_SELF == GetArea(oPC);
    validTime= GetIsDawn() || GetIsDay() || GetIsDusk();

    if(validTime && bIsPCInArea)
        DestroyDrowItems(oPC);
}

void DestroyDrowItems(object oPC)
{
    object oDrowItem;

    oDrowItem= GetItemPossessedBy(oPC, "CF_DROW_ITEM");
    if(GetIsObjectValid(oDrowItem))
    {
        SetPlotFlag(oDrowItem, FALSE);
        DestroyObject(oDrowItem);
    };
}
```

The script first checks for the time of day, retrieves the PC object, and retrieves the current area of the PC object. If all conditions are true, then the PC is searched for the amulet using **GetItemPossesedBy( )**. If the amulet is found, it is destroyed. This was done in the **OnHeartbeat** script handler because, unlike an **OnEnter** script, it handles the case where the PC enters at night but then lingers until daytime.

An embellishment would be to then give the PC a new item—a pile of dust—after the destruction of the amulet.

Chapter 4: Objects, Encounters, and Triggers

In the last chapter, we examined the properties and script handlers of the module and its areas. In this chapter, we bite off a bigger chunk:

- **Placeable Objects:** Simple objects (such as barrels, signs, and bird cages), and special case objects, such as doors.
- **Encounters:** Bound areas, usually triggered by the PC, that automatically spawn in one or more NPCs or creatures (usually hostile to the PC).
- **Triggers:** Three types of triggers are available: area transitions, generic triggers, and traps.

All of these objects combine to form the meat of the module. They provide atmosphere and opportunities for interaction. They are the props in the stage sets of your grand campaign.

1 Placeable Objects

We covered how to use placeables to add scenic flavor to your areas in Chapter 1. But placeables can also be used as repositories for treasure, as triggers for traps, and as elements of complicated puzzles. To make something more out of a placeable, right-click on it and bring up the **Placeable Object Properties** window.

Basic Object Properties

Name: The name of the object, which determines how it will be ordered in the Module Contents window, and what name will appear when it is examined by a player.

Tag: The object's internal name, which you would use when referencing that object in a script.

Appearance Type: Use this to make your object look like some other object. This is useful when you want to use the default values of one object and the appearance of another.

Plot: Check this box to make your placeable an indestructible Plot item.

Useable: A Useable item glows when the cursor is placed over it, indicating that it can be interacted with in some way (usually because it has inventory, but it could also serve as a switch with some scripted effect, a source of conversation, or a trap). All items in the Container list have Useable checked as a default, but all other items do not. Any item can be made Useable.

Has Inventory: Click this box (only available for items that have already been made Useable), and you'll be able to add inventory to your object.

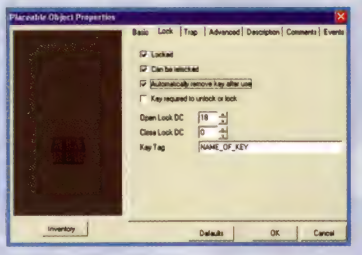
Hardness: This is what items have instead of Armor Class. The Hardness value of an item is subtracted from the damage roll of its attacker. So if you do 8 damage to a chest with a Hardness of 5, it takes 3 damage.

Hit Points: This works the same as it does for creatures and players.

Saving Throws: I can't imagine why you'd want to change these. Are your PCs ruining your campaign by casting Finger of Death on crucial armoires? Then up their Fortitude saves. Does your storyline demand an unusually agile Fenberry Bush that's capable of dodging fireballs? Send the Reflex save through the roof.

Lock Options

The Lock menu only appears if you make an item Useable.



Locked: Click this to bar this door to players without a key or the Open Lock skill.

Can Be Relocked: If clicked, a player can choose to relock the item afterwards. This could have interesting implications in online games where players are in competition.

Automatically Remove Key After Use: If the player has the key in his inventory, this will delete it from the inventory upon unlocking it. There's no reason to make PCs lug around keys to doors that they've already unlocked, unless the key opens multiple doors or there's a chance some other player could have relocked it.

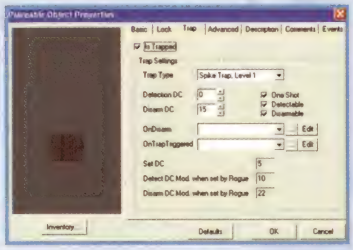
Key Required to Unlock or Lock: Only a key can open this door—no lock picking allowed.

Open Lock DC: The Difficulty Class to Open Lock this door. Remember that outside of combat, PCs can always take 20 when picking a lock. So a DC lower than 20 won't keep out even the most inept rogue. Think of what level character the area is intended for when you set these; even with Dexterity bonuses and lock picks, most non-rogues won't be able to unlock anything with a DC higher than 30.

Close Lock DC: The Difficulty Class to relock the door.

Key Tag: Once you've created your key as an item, simply enter its tag here. That's all it takes to set up a key lock on a door—no scripting required.

Trap Options



The Trap menu only appears if you make an item Useable.

Is Trapped: Nothing here matters if you don't check this box.

Trap Type: This determines what nasty thing happens to players when they open your trapped door.

Detection DC: The Difficulty Class for noticing the trap. Players make a Search check that has to beat this number to notice the door. Only rogues can detect traps with a DC of 25 or higher.

Disarm DC: The Difficulty Class to deactivate the trap with a Disable Trap check. Attempting to remove the trap (so it's added to your inventory) adds 10 to this number. Only rogues can disable traps with a DC of 25 or higher, which puts world builders in a bit of a pickle. Less than 25, and it's a snap for rogues; more than 25, and you're dooming non-rogues to setting it off.

One Shot: A One Shot trap is deactivated after it blows up in your face. If you don't check this, it will go off every time you try to open the object.

Detectable: Unchecking this box makes an object undetectable. This is a good way to infuriate your players.

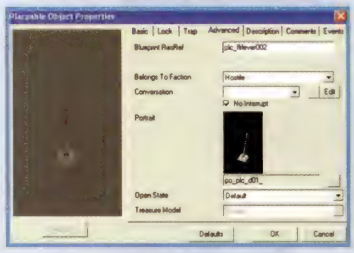
Disarmable: Unchecking this box makes an object impossible to disarm. This is an even better way to infuriate your players.

OnDisarm: This is where you can enter custom scripts that are fired when the trap is successfully disarmed.

OnTrapTriggered: If you want the explosion to do something above and beyond the chosen effect, like summoning guards or locking down an area, this is where you put the script.

Set DC & Other DC Mods: These fields show the stats of the trap that you've selected under Trap Type, so you know what dangers NPCs or competing PCs face if the trap is recovered and reset.

Advanced Object Properties



Blueprint ResRef: An object's tag is its name as an individual instance of that object. The Blueprint ResRef is the name of the object template. If you wanted to make a custom placeable, like a fern that is generated in a certain place at a certain time of day, the script that generates the fern would create it by calling its Blueprint name, not its Tag.

Belongs To Faction: I find it deeply unsettling that all the furniture defaults to Hostile... Does BioWare know something I don't? Is my coffee table plotting my death? Of course, inanimate objects lack the ability to do anything to the player, so the faction here is probably meant to be the faction of the object's owner. This could be used in scripts where a ShoutDisturbed function alerts NPCs of the same faction to the fact that the chest is being plundered.

Conversation: You can give a Conversation script to any object, creating the illusion of a sinister sentient shrub or a talking parrot in your bird cage. This is also where descriptive text that is called by an event would be placed, like "the rusty hinges creak loudly when you open the chest."

No Interrupt: If your enchanted lectern is a pivotal character in your campaign, check this box to make the conversation uninterruptible.

Portrait: If you change the Appearance Type of your object, the Portrait changes automatically. Hit the "..." box to select a new portrait for your object.

Open State: This useful option allows you to make your objects dead on arrival, open, closed, etc., depending on the object. Before you set this, right-click on the object, choose Animate, and select the chosen state to make sure it's one that's supported. This automatically updates the Open State field in the Advanced menu.

Treasure Mode: This pull-down menu allows you to change the default treasure-generating scripts without having to do it manually in the Events tab.

Other Object Properties Tabs

Description: Change the description that players view when they examine your object. By hitting the "..." button, you can change the description in various languages.

Comments: These are only for your edification. These will appear in the Object Preview window, so when you have a lot of chests you need to keep track of, making a note (as BioWare did in the official campaign) of the treasure type and lock and trap status is very convenient.

Events: This tab has a variety of hooks that can be used for scripting. More on that next.

2 Placeable Object Scripting

Due to the large cross-section of the *Neverwinter Nights* Toolset covered in this section, we'll focus on the new aspects of each object and discuss the common features they share. For example, the **OnEnter** event handler is present in the module, area, door, encounter, and trigger objects. Since the functionality is the same for all cases, we'll only discuss new event handlers and features.
There are 12 event handlers for a standard placeable. They are grouped as follows:

Opening/Closing/Locking—OnOpen, OnClose, OnLock, and OnUnlock

Damage Related—OnPhysicalAttacked, OnDamaged, and OnDeath (which activates when the object is destroyed)

Interaction—OnUse, OnDisturbed, and OnSpellCast

Custom and Heartbeat—OnHeartbeat and OnUserDefined

Scripting the Opening, Closing, and Locking Events

The **OnOpen** and **OnClose** event handlers fire when you would expect: whenever the object is opened or closed. In the official campaign, OnOpen is used most often for generating treasure.

Treasure Generation

When you generate an object from the Container list, it may have a default class treasure script in the **OnOpen** handler (and the same one in the **OnDeath** handler, so the treasure will appear whether the chest is opened or destroyed). Without doing any scripting yourself, you could substitute the default script for one of the many other treasure generating scripts created by the official campaign. All of the treasure generated by these scripts will be appropriate to the level of the character (so low-level characters have virtually no chance of getting high-value items, and high-level characters do not get junk from high- and boss-quality chests).

nw_o2_generallow (also **nw_o2_generalmed** and **nw_o2_generalhig**)—Generates a small amount of gold and/or general purpose treasure, with both the quantity and quality improving in the medium and high versions.

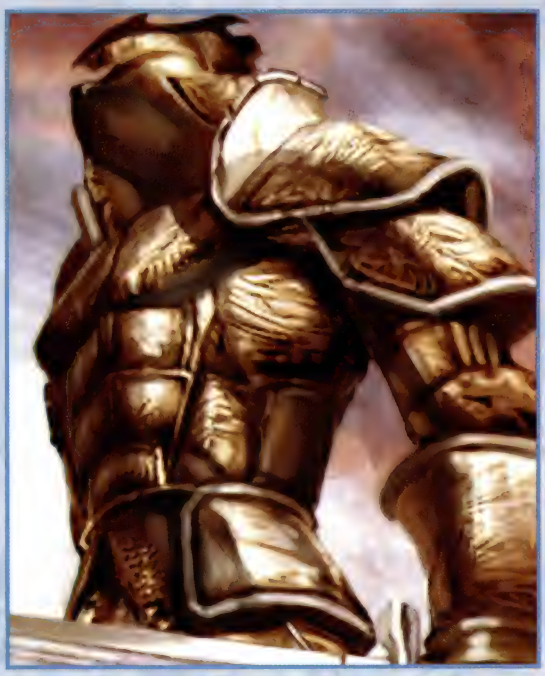
nw_o2_classlow (also **nw_o2_classmed** and **nw_o2_classhig**)—Generates a small amount of treasure that is usable by characters of the class(es) of the PC opening the chest, with both the quantity and quality improving in the medium and high versions.

nw_o2_boss—Like the class scripts, this always generates class-specific stuff, but is weighted more heavily toward weapons, armor, and magic items. Since these chests only appear after bosses in the official campaign, they usually contain the best stuff.

nw_o2_feat—This nifty scripts goes one better than the class-specific scripts and scans the chest-opening character for the Weapon Focus feat. If it finds one, it generates a random weapon of that type (the quality is still based on character level).

nw_o2_bookshelf—This script, which is the default for bookshelf and book pile placeables, generates random treasure only from the book list (70% chance), arcane scroll list (20% chance), or divine scroll list (10% chance).

nw_o2_coninclude—This is the script with all the treasure generation tables, and it defines the functions used by the above scripts. If you wanted your module to generate some of your custom items (like user-made books about your area) you'd need to wade in here and make your own version. Otherwise, leave it alone.



The following script (**nw_o2_generalhig**) demonstrates the contents of a typical treasure-generation script.

"NW_O2_GENERALHIG" Script Code

```
#include "NW_O2_CONINCLUDE"


void main()


{
   if (GetLocalInt(OBJECT_SELF,"NW_DO_ONCE") != 0)
   {
     return;
   }
   object oLastOpener = GetLastOpener();
   GenerateHighTreasure(oLastOpener, OBJECT_SELF);
   SetLocalInt(OBJECT_SELF, "NW_DO_ONCE",1);
   ShoutDisturbed();
}
```

The first line prevents the container from becoming a bottomless chest of treasure by checking the current value of **NW_DO_ONCE**. If the chest was never opened, **NW_DO_ONCE** is equal to 0 and the remaining code executes. However, any other value immediately ends the script, returning to the game. Let's assume that this is a fresh chest. The next line calls a function from **nw_o2_coninclude**, **GenerateHighTreasure( )**. This function generates expensive treasure, based in the current level of the PC (passed as the first parameter) and sets **NW_DO_ONCE** to 1.

The last line is interesting (and in every default treasure script). This is another **nw_o2_coninclude** function that makes NPCs go hostile when a chest that shares the same faction is destroyed. For instance, there is an inn filled with commoner faction NPCs inside. There is a chest in one of the rooms with the same faction. When the chest is destroyed and the contents taken, the commoners in the inn become (justifiably) hostile.

OnClose, OnLock, and OnUnlock Scripting

Possible uses for the **OnClose** script include: triggering a custom event based on the current contents of a chest, destroying any objects placed inside, and replacing the contents with a magical item (representing a magical assemblage of the previous items).

The **OnLock** and **OnUnlock** scripts are called when the object is locked or unlocked. Maybe the noise of the lock-pick sends the owner of the object to investigate, or maybe unlocking the object causes a secret door to open or passage to be revealed. Of course, the obvious nasty event is to pop the little thief who's fiddling with the object in the face with some poison or magical effect... a little blindness perhaps?

Damage-Related Scripting

These scripts deal with physical and magical attacks on the chest. Common events that trigger these include: bashing the chest, attacking it with a physical weapon, or taking damage from a fireball that just exploded nearby.

One obvious application for **OnPhysicalAttacked** and **OnDamaged** scripts is to alert guards in another room or the owner of the object. What a nice surprise for the party when the Ancient Blue dragon from Chapter 3 finds out that nasty little thieves are running loose in his lair!

The **OnDeath** script may seem a little weird for objects. (When was the last time a dying barrel choked out a desperate plea of love for a beer stein before dying in your arms?) However, this represents the destruction of the object by physical or magical

means. By default, the same script provided for **OnOpen** is provided for **OnDeath**. This represents smashing open the chest to get at the goodies inside.

One nice application for this script is to destroy any fragile items (such as a Fireball scroll or a potion of Haste) if the chest was "broken open." But if the chest was delicately picked open by a thief or opened normally, the potion or scroll would be there along with the more durable treasure.

Interaction Scripting

These scripts deal with interaction of the object and the PC, such as "using" the container (accomplished by clicking on the object in the game), casting spells at it, or disturbing the contents.

The **OnUse** script is very useful for displaying in-game information or short text messages from the object being used. For example, you can place a sign object in the game and give it a short one-liner conversation (such as **"Arcane Magic Training. Instructor: Jaroo"**), and then write a script for the **OnUse** handler, like the following:

Example OnUse Script

```
void main()

{

  SpeakOneLinerConversation();

}
```

When the PC clicks on the sign to read it, the conversation snippet appears, informing the PC that the nearby door leads to magic training. It's also very helpful to use for debugging or testing your module. We used them all the time to display current variable values and to cause events to occur on demand.

The **OnDisturbed** script is called when a chest's contents are disturbed. For example, there are chests throughout Neverwinter that present a riddle when examined. If the PC figures out the riddle (in the case of our example, by placing any missile-type ammunition into the chest), the chest unlocks a nearby door. Kind of a password mechanism, but very cool when encountered in the game. The **OnDisturbed** script for the chest (**m1s1chest3_5**) does all the work.

Example OnDisturbed Script

```
void main()

{

  int nItemBase = GetBaseItemType(GetInventoryDisturbItem());


  if(nItemBase == BASE_ITEM_ARROW || nItemBase == BASE_ITEM_BOLT ||

    nItemBase == BASE_ITEM_DART)

  {

    object oDoor = GetObjectByTag("Wall1Gate");

    SetLocked(oDoor,FALSE);

    AssignCommand(oDoor, ActionOpenDoor(oDoor));

  }

}
```

In the script above, the base type of the item placed in the chest is retrieved. If it matches one of the types needed (basically, is it missile ammo?), the nearby door object is retrieved, then unlocked and opened. For another example, see the Disturbing the Dead sidebar below.

The **OnSpellCast** script is another hook that triggers when any spell is directed at the object. This might be helpful for notifying an NPC if the PC used a Knock spell to silently open the chest. It can also allow you to do fun things, like creating portals that can be destroyed by a Dispel Magic spell. We'll discuss that example in the Those Pesky Portals of Endless Death example on page 62.

Example: Disturbing the Dead

The crypt area in the Haven module is full of fallen Rivenvein heroes, buried with powerful items in their coffins. Whenever an item is taken, a guard monster (let's make it a Helmed Horror) appears in the room, and the gate to the next area closes and locks (although it can be opened by rogues). If the item is returned, the gate unlocks and opens. However, the summoned guard remains. A switch at the end of the hall will destroy the crypt guards, if you survive long enough to get there. The locks can be opened by a competent rogue, however, so they stand a small chance of making it out alive, *with* the treasure.

To accomplish this, we'll script **OnDisturbed** and **OnClose** events for the sarcophagi, place waypoints for the guards to spawn into, and script an **OnUse** event for the "kill switch." Because the behavior is very similar between all sarcophagi, we'll only look at one example. The remaining **OnDisturbed** scripts can be easily generated from this one, with a few modifications.

OnDisturbed Script

```
void SpawnHorror(location loc);

void main()

{

  object oRelic = GetInventoryDisturbItem();

  object oDoor= GetObjectByTag("CF_Door3");

  location oLoc1= GetLocation(GetObjectByTag("CF_WPSarc1"));


  if( (nRelic == GetObjectByTag("CF_Dagger")) &&

    (GetLocalInt(OBJECT_SELF, "CF_Taken1") == 0))

  {

    SetLocalInt(OBJECT_SELF, "CF_Taken1", 1);

    ActionCloseDoor(oDoor);

    ActionDoCommand(SetLocked(oDoor, TRUE));

    ActionDoCommand(SpawnHorror(oLoc1));


  }

}

void SpawnHorror(location loc)

{

  CreateObject(OBJECT_TYPE_CREATURE, "nw_helmhorr001", loc, TRUE);

}
```

This script first gets the item that triggered the event, the spawn-in location for the Helmed Horror, and the nearby door object (representing the only escape route). If the item is a relic (determined by comparing the relic tag with the tag of the object that disturbed the sarcophagus) and this is the first time the item has been taken, a few exciting events occur. First, a local variable is set to prevent the same thing from happening again. The door is then closed and locked. Finally, the guardian is brought in to protect the relic. The script can be modified for other sarcophagi by changing the location passed to SpawnHorror, the tag identity of the relic, and the local variable indicating if the item was taken previously.

The **OnClose** script is simpler. It checks to see if the relic was returned when the sarcophagus was closed.

OnClose Script

```
void main()
{
   object oPossesor = GetItemPossessor(GetObjectByTag("CF_Dagger"));
   object oDoor= GetObjectByTag("CF_Door3");

   if(oPossesor == OBJECT_SELF)
   {
      SetLocked(oDoor, FALSE);
      ActionOpenDoor(oDoor);
   }
}
```

The current possessor of the relic is retrieved and checked against the sarcophagus object. If a match is made, the door is unlocked and opened. Unfortunately, the Helmed Horror is still hanging around.

As with the previous script, this can be modified by changing the tag of the item being checked for each remaining sarcophagi.

The final script is the **OnUse** script for the "kill switch."

OnUse Script

```
void main()
{
   object curHHObj;
   int nNth= 0;

   curHHObj= GetObjectByTag("CF_HELMHORROR", nNth);
   while(GetIsObjectValid(curHHObj))
   {
      ActionDoCommand(ApplyEffectToObject(DURATION_TYPE_INSTANT, EffectDeath(),curHHObj));
      nNth++;
      curHHObj= GetObjectByTag("CF_HELMHORROR", nNth);
   };
}
```

The first existing Helmed Horror is retrieved and checked for validity. If it passes, it is destroyed and the next Helmed Horror is grabbed. The check-and-destroy process continues until all existing Horrors are destroyed.

OnHeartbeat and OnUserDefined Scripting

This differs little from the module and area event handlers. Use **OnHeartbeat** for scripts that need to check something repeatedly or to perform the same action regularly or several times. See the Poison Fog example in our module.

The **OnUserDefined** script is used for custom events that occur when a certain condition is met or when something unique or "scripted" needs to occur.

Example: Poison Fog

Because our Yuan-Ti are resistant to poison, they fill the tunnels with poison gas to keep uninvited guests out of their temple (we'll make that by filling the area with a thick purple fog). The alchemist in town knows an antidote: an amulet that makes you immune to that specific poison. If the PC does not have the amulet equipped, the poison fog (implemented with an invisible object) nails him for 1d4 damage per round. The damage is of type *magical* to avoid the Damage Resistance of the PC.

The easiest way of implementing this is to have the damage come from an invisible object placeable set somewhere in the poison tunnels area map. We added the following script to the **OnHeartbeat** event handler of the invisible object.

OnHeartbeat Script

```
void main()
{
 object oPC= GetFirstPC();
 effect eDamage = EffectDamage(d4(), DAMAGE_TYPE_MAGICAL, DAMAGE_POWER_PLUS_FIVE);
 object oAmulet= GetItemInSlot(INVENTORY_SLOT_NECK, oPC);

 if(GetIsPC(oPC) && (OBJECT_SELF == GetArea(oPC)))
  if(GetIsObjectValid(oAmulet) && (GetTag(oAmulet) == "CF_POISON_AMULET"))
    return;
  else
    ApplyEffectToObject(DURATION_TYPE_INSTANT, eDamage, oPC);
}
```

The script first declares/initializes a few needed variables and retrieves the PC object, using **GetFirstPC( )** and whatever item is in the **INVENTORY_SLOT_NECK**. The current area of the PC object is then checked. If the PC is in the poison tunnels area and the amulet is equipped, the damage effect is prevented. However, if the PC is not wearing the amulet, the PC gets 1d4 upside the head every 6 seconds! Note that to keep this example simple, we made it so that it would affect only a single PC. To make it affect a party of players, you can add a simple while loop with the **GetNextPC** function.

This is a good example of what the Invisible Object placeable (under Miscellaneous) is for. If you make your Invisible Object into a Plot object that is not Useable, no one will ever know that it's the actual source of the damage.

3 Door Objects

Doors have a list of properties similar to objects. Their Basic, Advanced, Lock, and Trap settings are identical, with the exception that doors cannot have inventory and are always Useable.

Note

To make a door simply be scenery, set it to Plot, and Lock it with a DC of 100. An OnUse or OnFailToOpen conversation script that says "this door is barred from the other side" or "there doesn't seem to be any way to open this door" will keep your players from wasting time trying to open a door to nowhere.

What makes doors different is the Area Transition tab. In fact, a door is really just a placeable with a built in Area Transition Trigger. Like any Area Transition, a door can be connected to another door, a waypoint, or an Area Transition trigger. To do a simple one-way transition, you can input the tag of the door, waypoint or trigger in the Destination Tag field… And you're done. To make it a two-way door, input the original door's tag in the Destination Tag field of the second door. To automate the process, hit **Setup Area Transition**.

The Setup Area Transition editor lets you select your destination area from a pull-down menu. Then chose your target type: Door, Trigger (Area Transition), or Waypoint. A list of possible targets are displayed, both in a list and on the map, and you can simply click the one you want, choose the Connection Type, and hit OK. You can even select the loading screen that will be displayed to players from a list of available bitmaps.

Door Scripting

There are 10 standard event handlers and 2 unique event handlers for a standard door:

Standard Event Handlers—OnClose, OnDamaged, OnDeath, OnHeartbeat, OnLock, OnPhysicalAttacked, OnOpen, OnSpellCastAt, OnUnlock, and OnUserDefined

Unique Event Handlers—OnAreaTransitionClick and OnFailToOpen

The standard event handlers work in the exact same way they did for placeables, so see the previous section for details on those. We'll go right to the unique event handlers.

Door-Specific Scripting

By default, the **OnAreaTransitionClick** event handler for a door object is empty. This causes the default transition behavior to occur, jumping the PC to the destination location you set up in the Area Transition tab of the door object. *However, when you add a script to this event handler, you destroy the previously set-up behavior.* This means that if you put something in this handler, you must manually script the transition. For more information, see the Area Transition-Related Scripting section at the end of this chapter.

The **OnFailToOpen** script fires when the character attempts to open a locked door. This is useful for scripting interesting events like a prisoner saying something when the PC attempts to open their cell door, or telling the PC something important about the door (like it's magically warded and cannot be opened without a key). There is an interesting **OnFailToOpen** script in the Peninsula District, covered in Chapter 1. An escaped prisoner on the other side of the door taunts you when you fail to open the door. That's easily set up with a simple **ActionStartConversation** function. But what if you had already looped around to the other side and killed that prisoner? It's unlikely anyone will try it, but BioWare is ready if someone does.

Example OnFailToOpen Script

```
void main()

{

 object oPC = GetEnteringObject();

 object oDhanis = GetObjectByTag("M1Q02Cdhanis");

 object oInteriorDoor = GetObjectByTag("m1q2B_m1q2A");

float fDistance = GetDistanceBetween(oDhanis, oInteriorDoor);
```

```
if (!GetIsDead(oDhanis))
{
      if (GetIsObjectValid(oDhanis))
      {
            if(GetIsObjectValid(oInteriorDoor))
            {
                  if (fDistance < 15.0)
                  {
                        ActionStartConversation(oPC);
                  }
            }
      }
}
}
```

Now that's thorough! Before the conversation begins, this script checks to make sure Dhanis (the prisoner) is alive and valid, that the door is valid, and that Dhanis is within 15 feet of the door. Otherwise, you get nothing but a cold, silent door.

Example: I Hear You Knocking, But You Can't Come In

When you see a door that's barred from the other side, that's usually the designer's way of saying "there's no way you'll ever be able to open this door, so move on." But what if you actually wanted to implement the barred-from-the-other-side mechanic? In our module, players are forced to sneak into a sealed vault through an underground crypt, because the door has been barred by survivors on the inside. Once they've gone the long way around, they should be able to lift the bar and leave through the front door.

To implement this, we made the vault door a Plot object, and set its **Open Lock DC** at 100. When the PC tries to open the door, a message displays, indicating that the door is barred. The door becomes unbarred if the PC opens the door from the underground crypt side. This requires two scripts: one on the **OnFailToOpen** handler of the door that displays the appropriate message, and another for the OnOpen door of the crypt side that unlocks the originally barred door. Here is the **OnFailToOpen** script, used in the Haven module:

OnFailToOpen Code

```
void main()
{
  SpeakString("This door is barred from the other side.");
}
```

A blindingly simple script, it presents the current state of the door. However, the other part of the solution is the unlocking of the door from the crypt side.

This is handled by a script for the **OnOpen** event handler of the crypt side door.

OnOpen Script

```
void main()

{

 object oDoor= GetObjectByTag("CF_VaultDoor");


 SetLocked(oDoor, FALSE);

 ActionOpenDoor(oDoor);


}
```

In this script, the vault door object from the related area is retrieved. Then the door is unlocked and opened. Therefore, when the PC transitions through the door (or approaches the vault door from the other side) it will seem as if the door has been unbarred and opened.

4 Encounters

An encounter is a trigger that automatically spawns in enemies. The advantage of using an encounter, as opposed to simply having enemies standing around, is that the encounter can check the power level of the PCs in the party and spawn in a level of opposition that will always be consistent in difficulty. It also saves the software from having to keep track of lots of enemies by not bringing them into existence until the player is about to encounter them, and allows you to set up surprise attacks where enemies spawn in all around you.

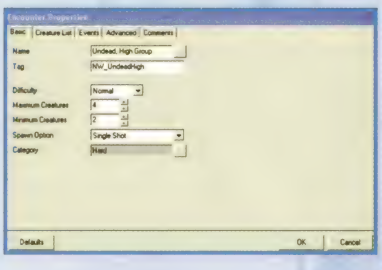
Scalable Difficulty in Your Module

It can be difficult for module designers to decide when to use encounters and when to use preset NPC enemies. The simplest way is to use preset NPCs for all your major battles and encounters for what are called "random encounters" in pencil-and-paper *D&D*. (Random encounters are randomly generated battles that exist to keep players interested and gain experience, even though they do nothing to advance the plot.) But if you're concerned about making your campaign equally challenging for characters with a wide range of levels, you can even make your bosses as encounters. Just make your boss as a custom NPC template, and copy and paste it, making several versions with the same conversation file and scripts, but different stats that affect the CR level. Then make a custom encounter with the different versions of the boss as the only possible enemies, and tell it to spawn in only one creature. Set the spawn point where the boss would be, and put the trigger in a place where players have to step on it, or have it envelop the entire area. Now you can change the difficulty of the encounter simply by selecting it in the Difficulty pull-down menu.

This is a very user-friendly way of making an adventure, and if you put enough time and energy into it, you can make an adventure that will be equally challenging to characters from levels 1 to 20. But this method has its downsides, too. There's something very contrived about going through an adventure always running into enemies who are just slightly stronger than you are, no matter how powerful you are. And when players figure it out, they'll have no incentive to gain experience. Fixed encounters give players the goal of becoming strong enough to, say, take out that high-level Lich in the official campaign. And one of the simple joys of gaming is going up a few levels, returning to an area that used to give you fits, and effortlessly shredding the enemies like field mice in a rotary mower.

So here's a rule of thumb for module builders who are willing to take the time to make their quests entertaining for characters with a wide spectrum of experience: If it's a battle that players have to fight to proceed, but you think it could be too difficult for low-level characters, make it a scalable encounter. But if it's an optional side quest, use some preset NPCs to keep parts of it at a fixed difficulty level.

Basic Encounter Properties



Name: The name of the encounter.

Tag: The encounter's internal name, which you would use when referencing the encounter in a script.

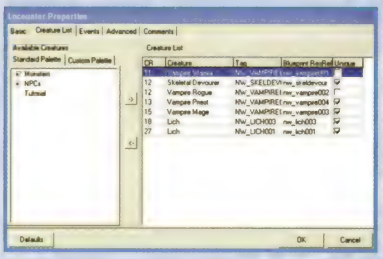
Difficulty: Determines how difficult you want the encounter to be. This is constrained by the monsters in the Creature List; if your monsters only range from CR 5 to 8, setting it to Impossible will never make it challenging for level 20 characters, and setting it to Very Easy won't make it possible for Level 1 characters.

Maximum/Minimum Creatures: One of the ways to extend the difficulty range is by putting a wide selection of values here. For example, even if your Creature List consists of only CR 8 creatures, you can set this from 1 to 8 (the max). Eight CR 8 creatures will be tough for a level 12 character, and a level 3 character has a fighting chance against a single CR 8 foe. But for flavor reasons, you may want the enemies to appear only in small groups or large armies.

Category: The difficulty category where you've placed this in the palette. For your reference only.

Encounter Creature List

This is where you choose the monsters you want to include from the Standard and Custom Palettes on the left. When you find one you want to use, select it and hit the arrow button to put it in the Creature List. If you want to make a monster unique for flavor reasons (a single Lich leading a pack of undead minions makes more sense than a pair of liches), click the Unique box.



Advanced Encounter Properties

Faction: Hey, your encounters don't have to be enemies. You can set up a trigger that spawns in a bunch of defenders. Or make the enemies that are of different factions that both hate the PC, but also hate each other, allowing interesting strategic possibilities for players who want to stand back and let the monsters kill each other. Note that changing this won't change the factions of the creatures spawned in; it will just help you categorize them, and could be used in scripts that fire encounters of a certain faction.

Active: If this isn't checked, the encounter won't work. You may want to make inactive factions and then have a script that activates them at some point in the quest (say, an Area OnHeartbeat script that activates werewolf encounters at night).

Encounter Respawns: Most encounters are one-shots, but encounters could be made to respawn a fixed amount of times, or even infinitely. This is useful for those frequently traveled wilderness areas between two towns, since you don't want the players to be bored out of their minds every time after the first that they cross through there. An encounter can respawn even if the creatures it previously spawned in are still alive, so to create a generator that keeps spitting out enemies, set the Respawn Time to a small number and put it on **Infinite Respawn**.

Player-Triggered Only: As if henchmen don't get you into enough trouble already! That's one of the perils of turning this option off, but it also lets clever players trigger ambushes with their familiars, and allows you to set up events where NPCs stumble into hostile ambushes.

5 Encounter Scripting

There are four standard event handlers and one unique event handler for an encounter object. They are grouped as follows:

Standard Event Handlers—OnEnter, OnExit, OnHeartbeat, OnuserDefined

Unique Event Handler—OnExhausted

Encounter scripting is fairly straightforward because there is only one unique event handler out of five. The four standard handlers handle any events or effects you want when the PC enters or exits the encounter area. The **OnHeartbeat** and **OnUserDefined** handlers allow you to regularly check for certain conditions (or cause others), and fire custom events from the encounter object or other objects in the module.

The unique event handler, **OnExhausted**, relates to the party spawned in by the encounter object. When the last member of the spawned-in encounter party is killed, the **OnExhausted** event handler is triggered. This can be used to open a previously locked door or trigger some other event once the PC has vanquished the encounter group.

Example: Those Pesky Portals of Endless Death

Here's an idea: We want a portal that keeps an encounter active until the portal is destroyed. The PC can destroy the portal by casting Dispel Magic or a similar spell at the portal. Not a spellcaster? Then the PC can run.

Implementing this will require two things:

1. A script for the **OnSpellCastAt** event handler of the portal that checks to see if the spell cast was Dispel Magic (for the sake of this example, we'll make that the only effective spell). If the spell was Dispel Magic, the portal and encounter objects are both destroyed.

2. An encounter that has the **Encounter Respawns** and **Infinite Respawn** options selected. The spawn point of this encounter will be set at a portal placeable, giving the impression that the portal is doing the spawning.

Here is the script for the **OnSpellCastAt** handler.

OnSpellCastAt Script

```
void main()
{
 object oEnc;

 if(GetLastSpell() == SPELL_DISPEL_MAGIC)
 {
  oEnc= GetObjectByTag("CF_Goblinoid");
  if(GetIsObjectValid(oEnc))
  {
      ActionDoCommand(DestroyObject(oEnc));
      ActionDoCommand(SetPlotFlag(OBJECT_SELF, FALSE));
      ActionDoCommand(DestroyObject(OBJECT_SELF));
  }
 }
 else
  return;
}
```

Because this is an **OnSpellCastAt** event handler, we can use **GetLastSpell( )** to retrieve the last spell targeted at the object—in this case, the portal. If that spell was Dispel Magic, then we retrieve the infinite respawn encounter and destroy both the encounter object and the portal object. The design of this solution is very similar to the activation and deactivation of encounters based on the status of a specific quest, discussed in Chapter 3.

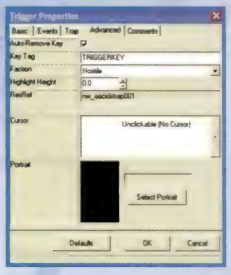
6 Triggers

There are three kinds of triggers:

- **Area Transitions** are exactly like doors, they just aren't stuck in doorways. These are typically used at the edge of areas that continue into other areas... Just paint a large rectangular Area Transition on both sides and connect them. Area Transitions are connected in the same manner doors are connected.
- **Generic Triggers** are used for user-scripted events that you want to fire when the player steps into a specific area. These are generally invisible to the player (unlike Area Transitions which glow blue, and Traps which can be detected), and are generally used for custom traps (step here to activate the golems), opening doors, or initiating conversations from NPCs (step here and Pavel comes running).
- **Traps** are the ever-popular floor traps that activate when stepped on. You choose the Trap Type and its various properties and DCs the same way you would for a door or object.

Advanced Trigger Properties

Most of the Trigger Properties you've seen before. But there are a couple of interesting things you can do in the Advanced Trigger Properties tab:



- **Key Tag**: Yep, you can designate a key on a Trigger. If a player steps onto the trigger while holding the key, the trigger won't fire. You could give a player a Ward Stone that makes him or her immune to the traps in an area, for example. To make the key a one shot effect (get out of one trap free!) check the Auto-Remove button.
- **Cursor:** If your script has an event in the OnClick event handler, you can choose what the cursor looks like when you hold it over the object here.

Trigger Scripting

There are four standard event handlers and one unique event handler for all trigger types. They are grouped as follows:

- **Standard Event Handlers**—OnEnter, OnExit, OnHeartbeat, and OnUserDefined
- **Unique Event Handler**—OnClick (discussed in the Area Transition-related Scripting section).

Exactly like their Encounter event handler counterparts, these handlers fire when the PC object enters or exits the trigger area, receives a custom event signal, or at each "heartbeat" event (every 6 seconds). However, triggers can also be made to react to the player clicking on the trigger area. This is used in Area Transitions, which we'll discuss at the end of this chapter.

Example: The Dead Shall Rise Again

The Yuan-Ti temple is full of corpses, victims of the vivisection experiments tirelessly performed by their captors. Long numbed to the sight of corpses in our module, the player wanders calmly in, hitting an invisible trigger that causes the group of corpses, captives who had been turned into deranged cursts by the Yuan-Ti, to rise and attack the PC.

The solution has two parts. First, we need to place a generic trigger anywhere within the immediate area of the corpses. This trigger has a script (called **cf_trigger1_rm1**) for the **OnEnter** event handler. The script follows on the next page.

OnEnter Script

```
void main()
{
   object curZObj;
   int nNth= 0;

   curZObj= GetObjectByTag("CF_Curst3" , nNth);
   while(GetIsObjectValid(curZObj))
   {

ActionDoCommand(ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectResurrection(),curZObj));

ActionDoCommand(ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectHeal(GetMaxHitPoints(curZOb
j)), curZObj));
    nNth++;
    curZObj= GetObjectByTag("CF_Curst3", nNth);
   }
}
```

This script gets the first curst in the group and, if valid, raises and heals it completely. It then grabs the next one and repeats the cycle, until all the cursts are shambling toward the hapless PC. This script assumes that the cursts are killed in a certain way, and that they all belong to the same template. This example shows how to make them rise. In the next chapter, we will go into more detail on how we made the undead and the script that made them initially play dead instead of attacking the player.

Trap-Related Scripting

Trap scripting is a little different than general trigger scripting. The difference is in the **Trigger Properties** dialog for the Trap trigger object. Basically, there is a **Trap** tab containing two trap-specific event handlers: **OnDisarm** and **OnTrapTriggered**. I won't insult your intelligence by describing when these handlers are triggered. We'll just jump right into talking about each of them.

Several possibilities exist for the **OnDisarm** event handler:

- Opening previously locked doors in the room where the disarmer is currently trapped.
- Updating an existing quest, in the journal of the disarmer, that requires the disarming of traps in a certain order, or just requiring the trap to be disarmed.
- Perhaps manipulating the inventory of the disarmer by taking something from or placing something in his inventory. Perhaps the trap had an important part that is needed in another part of the module?

The **OnTrapTriggered** handler can be quite fun also. Some other possibilities include:

- Alerting an NPC (or several of them) that someone is sneaking around their area.
- Creating objects that are "parts" of the sprung trap for usage later.
- Causing the appearance (perhaps a mage teleporting in) of an unexpected guest.

Area-Transition-Related Scripting

Area transition triggers are clickable and viewable objects in *NWN*. By default, the **OnClick** and **OnEnter** handlers are empty, but the Area Transition will still jump the PC to a predetermined destination location when clicked or entered, if one was set up

beforehand in the Area Transition tab of the trigger object.

You can specify a script, for either **OnClick** or **OnEnter**, that customizes the behavior of either handler in many ways, but then you must also manually implement the transition portion in your custom script. Once a script is loaded into either handler, the default behavior is lost for that handler. The following script (Chapter 1 module, **at_areatrans_001**) shows how to handle this customization.

Example OnClick Script

```
void main()

{

 object oClicker = GetClickingObject();

 object oWaypoint = GetTransitionTarget(OBJECT_SELF);

 location lLoc = GetLocation(oWaypoint);

//

//Perform custom actions here

//


 SetAreaTransitionBMP(AREA_TRANSITION_CITY);

 AssignCommand(oClicker, JumpToLocation(lLoc));

}
```

This script first retrieves the object that clicked the trigger; presumably the PC. The transition target is retrieved (using **GetTransitionTarget( )**) as well as the location of that target (using **GetLocation( )**). Any custom scripting is then performed. Finally, a transition bitmap is specified and the **JumpToLocation( )** command is given to the clicking object. This jumps the PC to the destination of the area transition.

Did you notice that the same object that was clicked is used as the destination? It looks like the entering object would be transported back into the area it just left. Actually, the call to GetTransitionTarget is retrieving the existing destination of the Area Transition object—the one set up using the Properties dialog of the trigger object.

Example: Area Transition Shenanigans

During a climactic battle between two high-level mages in our module, the player is teleported away by an NPC. Upon returning to the same room (using the same door from before the fight), the player finds that it has been turned into a smoking wreck.

To accomplish this, we made two versions of his room, one normal and one destroyed. The door object connecting the room to the rest of the module can take the player to either room. The actual destination is determined by a local variable on the door object and a custom **OnAreaTransitionClick** script. This variable initially carries the tag of the first room. Upon some later action, the value of that variable changes to the tag of the second room. Therefore, when the PC walks through the door, after the teleportation effect, the actual destination has switched from the first room to the smoking wreck.

Assume that the variable (called **CF_CUR_WP**) is initialized in the **OnEnter** script of the area containing the door to Lord Neven's chamber. For example:

```
SetLocalString(GetObjectByTag("CF_DOOR"),
"CF_CUR_WP", "CF_WP_ROOM1");
```

The custom script for the **OnTransitionAreaClick** handler of the door object is as follows:

OnTransitionAreaClick Script

```
void main()

{

   object oClicker = GetClickingObject();

   if(GetIsPC(oClicker))

   {

      string newTarget= GetLocalString(OBJECT_SELF, "CF_CUR_WP");

      object oTarget2= GetObjectByTag(newTarget);

      location lLoc2= GetLocation(oTarget2);


      AssignCommand(oClicker, JumpToLocation(lLoc2));

   }

}
```

In this script, the clicking object (the PC) is retrieved. After verifying that it is the PC, we retrieve the current value of the **CF_CUR_WP** variable. Using the tag, we retrieve the current transition target object and get the location of that object. We then assign a **JumpToLocation** command to the PC, whisking him away to whatever room we desire.

This is a powerful trick that can be used anytime you need an area to change drastically. Since there is no way to script a tile change, if your module calls for players to return to an area that has been devastated, or rebuilt, or visibly changed in some way, you need to pull this little switcheroo. Just copy the area, change it in the toolset, and set up a script like the one above that will take them to the new version at the appropriate times.

One warning: Remember that the copied area is an entirely different area as far as the game is concerned. So chests that have been plundered will be full again, enemies that have been slain will be alive again, and encounters that have been triggered will be active again. Make sure you don't leave in anything that will give away the fact that the player is actually in a whole new area.

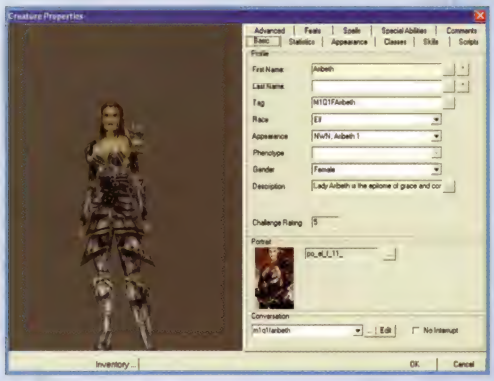
Chapter 5
Creatures, Conversations, & the Journal

1 Creature Properties

Creatures are one of the most important parts of any module. They're the NPCs that give you the quest, the henchmen that aid in its completion, and the monsters that stand in your way. An adventure without them would have no combat and no story, and be incredibly lonely.

The egalitarian toolset doesn't differentiate between NPCs and monsters; they're all just creatures in its eyes. They work the same way, and use the same scripts. Let's take a look.

Basic Creature Properties



First/Last Name: I can spend hours just hitting the name generator button. Wheee! Everything is organized by first name, so last names are usually just for flavor.

Tag: The internal name for this character, used frequently in scripts that affect an existing creature.

Race: Used in determining things like racial enemy bonuses and when called by script functions like **GetRacialType**. This has nothing to do with the appearance of the creature. For example, you can take a dragon, make its race Gnome, and it will no longer qualify as a ranger's dragon racial enemy, but will be vulnerable to weapons that do +5 vs. Gnomes.

Appearance: Select any number of preset creature models from this pull-down menu (the Portrait will also change automatically). If you want to create a custom appearance for an NPC instead of using a preset, set this to a basic humanoid race (Dwarf, Elf, Gnome, Half-Elf, Half-Orc, Halfling, or Human), and the **Appearance tab** options will be active. These are the only Appearance types that you can edit.

Phenotype: Another option that is active only for the above listed basic humanoid races. Select "Large" to make your NPC overweight.

Gender: For the above-listed basic humanoid races, this changes the appearance. The visual gender of other species can only be changed if they have an alternate gender form, although this field will still be accurately retrieved by a **GetGender** function, no matter what the species look like. Okay, I get having "None" as an option for Oozes and stuff, but "Other"? "Both"? Ewwww.

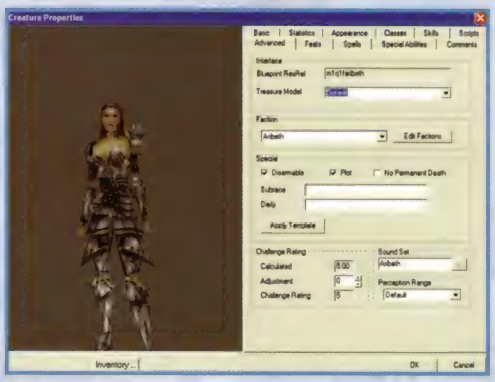
Description: This is what players will read when they examine a creature.

Challenge Rating: The CR of a creature affects what difficulty characters see (i.e., Challenging, Impossible, etc.) when they examine a creature, and also the amount of experience they get for defeating it. This is automatically recalculated as stats change.

Conversation: Here's where you input your brilliantly crafted dialog files.

No Interrupt: Check this box if what your character has to say is so important that you don't want to allow players to skip it with the Escape button.

Advanced Creature Properties



Blueprint ResRef: This is the name of the creature template, often used by functions that create the creature as part of an event script.

Treasure Model: At printing time, this was not yet implemented. Presumably, this lets you determine what treasure level a creature holds without a custom script.

Faction: The most important field for any creature. (More on this in the "Creature Factions" section of this chapter.)

Disarmable: Keep this checked unless the creature holds a weapon you absolutely will not allow to fall into a PC's hands.

Plot: The immortality box. If only it were this easy in real life…

No Permanent Death: Checking this box ensures that instead of fading out, the creature's body sticks around, allowing players to resurrect it.

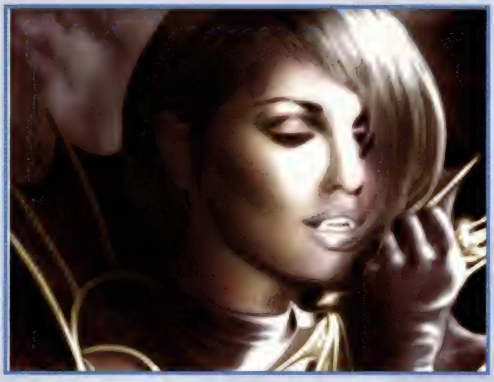
Subrace/Deity: In the official campaign, these do nothing. But for intrepid coders, this is the hook that allows you to script your own racial behavior. For example, you could make an Area **OnHeartbeat** script that uses the **GetSubrace** function to weaken all Drow during sunlight, but gives them special abilities when underground and at night. There's also a **GetDeity** function that you can use in similar scripts to grant special bonuses and penalties, or alter the reaction of other creatures to an NPC's Deity. Remember that players can choose their own Subrace and Deity when they make their characters, and GetSubrace and GetDeity can retrieve that information from players, too. Lots of cool possibilities here.

Challenge Rating: Challenge Rating doesn't take everything into account; your creature could have weak stats but custom scripts that give it incredible powers or a skin/hide that makes it immune to nearly everything. To make the XP characters receive reflect the creature's true power, adjust the CR here. Or you can just use this to screw PCs out of hard-earned XP.

Sound Set: A dazzling array of options.

Perception Range: Another very useful option when scripting creature behavior. You want the players to see the arrows before they see the elven snipers? Set the archers' perception range to "long." Want to make it easy for players to sneak past the elderly orcish jailer? Set his perception range to "short."

Other Tabs: You can read about how the basic stats work in 3rd Edition in the instruction manual. All you can do here is change base stats, and add bonuses to saves, AC, and hit points. This is the only place you can adjust a creature's **Movement Rate**, a fun statistic to play with when you're making custom vampire ninjas and whatnot.



The **Appearance** tab, as mentioned previously, can only be changed if your Appearance in the Basic Properties window is set to one of the seven basic humanoid races.

In the **Classes** tab, make sure to specify an Alignment (especially for creatures in combat), since a great many spells and abilities depend on this field. The classes can be retrieved by functions, but generally don't have a large impact on the play mechanics. **Skills**, **Feats**, **Spells** and **Special Abilities** (generally for monsters only) are all self-explanatory.

2 Creature Factions

The Elves and Dwarves get along well enough in our hypothetical town, but elves won't risk their lives to help a dwarf the way they will for a human. The humans are indifferent to half-orcs, but the dwarves hate them and are one incident away from starting a race riot. The lizardfolk in the swamp have a truce with the humanoids in town, but hate the dryads that the elves and humans fight to protect, leading to occasional hostilities. Imagine what a mess this would be to script without factions! But thanks to factions, you don't need to script anything to place the creatures in your module into various alliances and warring bands.

There are five standard factions:

PC: You can't edit the PC faction (since it's up to the players to decide how to behave), but how each faction feels about the PC determines who gets the Sword icon and who gets the Conversation icon when you put the cursor over them.

Hostile: Hostile creatures live to kill everyone, except other Hostile creatures. If you want two Hostile races that are hostile to each other (a likely scenario) you'll have to give them custom factions.

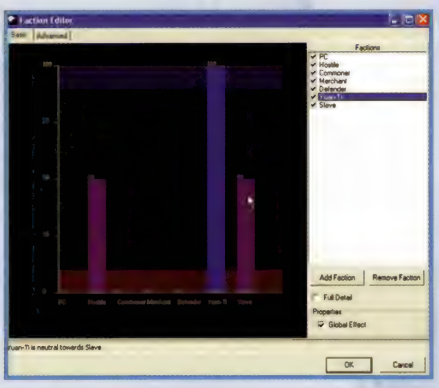
Commoner: Commoners are indifferent to everyone except other Commoners (whom they defend), and Hostile creatures (whom they hate).

Merchants: Merchants look out for each other and for Defenders. They hate Hostile creatures.

Defenders: Defenders also hate Hostile creatures, and risk life and limb to protect Commoners, Merchants, and each other.

The Faction Editor

Bring up the Faction Editor and take a look. It's all pretty simple: The red range is hostile, the blue range is friendly, and everything in between is neutral. All the possible factions are displayed in the faction list, and the ones that are checked are listed on the graph. You can then see the opinions of any one faction towards the selected factions by highlighting it in the list.



To make a new faction, hit "Add Faction." The parent faction is the one it will be based on; a child of the Hostile faction will have all of its default values the same as Hostiles (adjusting one after that point will not affect the other). This is handy when you have dozens of factions to keep track of, so it's a good idea to keep the standard factions updated as you add new ones. The **Full Detail** box will show you the relationship both ways, allowing you to see how the listed factions and the highlighted factions feel about each other. The **Advanced** tab will let you see the full chart of how every faction feels about every other faction, which is useful when you have too many to display in the Basic chart.

Using Factions

In the world of factions, everyone's opinion about everyone else can be depicted as a number between 0 and 100. If it's in the bottom 10%, you hate them enough to attack on sight. If it's in the top 10%, you'd lay down your life to protect them. The middle 80% is all the same, but it can be significant if your module will use the dozen or so script functions that adjust reputations. Say the Nixies in the swamp are indifferent to the party (50%), but a script uses the **AdjustReputation** function to alter their opinion based on certain events.

- If you rescue their lost princess, their opinion of the PC faction rises by 20%.
- For every lizardfolk enemy you kill, their opinion rises by 5%.
- For every nixie you kill, their opinion falls by 20%.

So if you save the princess and kill four lizardfolk, the nixies become powerful allies that fight by your side when they see you in trouble. If you kill two of them by accident or to steal their treasure, you'll have a new regional enemy on your hands.

Most faction usage is more mundane than that, of course. In our module, we couldn't have our Yuan-Ti slavers attack their own slaves, so we had to make them a new faction that is hostile to everyone except the Standard Hostile faction, and the Slave faction. The Slave faction was made to be indifferent to everyone. Ta-dah! Now the Yuan-Ti attack the player but leave their slaves alone.

There are about 50 different factions in Chapter 1 of the official campaign alone. Most of them are merely friendly to themselves, meaning they'll defend each other if attacked, but have no ill will towards anyone. So why have custom factions if they don't behave differently? One reason could be as a means of grouping creatures—halflings and gnomes may behave identically in your world, but when a scripted plague wipes out the Halflings and leaves the gnomes untouched, you'll be glad you gave them their own faction; the script can identify the halflings by faction and kill them all with just a few lines of code.

3 General Scripting for Creatures

The subject of NPC scripting is one of the largest issues facing a module designer. Because most modules are set in some type of populated area, a well-scripted horde of NPCs can really give your module life and atmosphere. Imagine a busy inn at a rural crossroads, filled with a handful of NPCs talking, socializing, drinking, and perhaps even a small halfling wandering about picking the pockets of the unwary.

BioWare was kind enough to provide several sets of scripts that provide basic behavior in a variety of situations, including combat and activities for day and night. There are two sets:

- **nw_c2_default*x***—A set of scripts for all 13 event handlers of a monster or NPC object, where *x* is 1–9 or a–e (except c)
- **nw_ch_ac*x***—A set of scripts for all 13 event handlers for the available henchman in the NWN campaign, where *x* is 1–9 or a–e (except c)

Note

The main NPCs for the *NWN* campaign are based on the standard NPC character models. However, due to their importance in the *NWN* campaign, some of their scripts are completely different from the default scripts provided for "common" NPCs and monsters.

It is left up to the module designer to tweak and enhance these scripts to turn their module into a living, breathing world with a special flavor of its own. In this section, we examine the default scripts provided for a creature in the game, and discuss the possibilities for modification. One final note before moving on: The default script for the **OnSpawn** event handler is by far the most influential default script in these sets. It can set a large number of variables, influencing all aspects of creature default behavior. Any spawn conditions discussed are set in this script. It makes a good beginning, even if it is out of order with the actual event handler list.

The OnSpawn Script

This event handler is the root of all default behavior for a creature. It is called when the creature object is created or spawned into the game. The default script does exactly three things:

- Sets the listening patterns for the creature
- Causes the creature to start patrolling or walking a specified set of waypoints, if available
- Generates a small amount of treasure for the creature

However, if you look at the actual script (called **nw_c2_default9**) you will notice that the entire script is over a page-and-a-half long and, for the most part, is commented out. This commented section is the key portion of the script. This script allows the user to set various conditions that access and modify the generic AI of the creature object. You simply uncomment the conditions you are interested in, and save the script under a different name. Once your custom spawn script is placed in the **OnSpawn** event handler, the generic behaviors have been altered.

There is a separate default script for the **OnSpawn** event handler of a boss monster. The only difference is that the boss scripts set a special spawn condition, called **NW_FLAG_FAST_BUFF_ENEMY**. If set, it causes the boss to create a fast defense by casting spells instantly before perception by the PC. More detail on this condition are presented later.

Let's examine the commented areas:

Optional Behaviors

This first section sets various behaviors for the Creature object that influence other creature default scripts. With some slight modifications on my part (for easier reading) they are as follows:

OnSpawn Script Optional Behaviors

```
// OPTIONAL BEHAVIORS (Comment In or Out to Activate)

*************************************************************************

// This causes the creature to say a special greeting in their conversation file

// upon Perceiving the player. Attach the [NW_D2_GenCheck.nss] script to the desired

// greeting in order to designate it. As the creature is actually saying this to

// himself, don't attach any player responses to the greeting.

     //SetSpawnInCondition(NW_FLAG_SPECIAL_CONVERSATION);

     //SetSpawnInCondition(NW_FLAG_SPECIAL_COMBAT_CONVERSATION);


// This will set the listening pattern on the NPC to attack when allies call

     //SetSpawnInCondition(NW_FLAG_SHOUT_ATTACK_MY_TARGET);


// If the NPC has stealth and they are a rogue go into stealth mode
```

```
      //SetSpawnInCondition(NW_FLAG_STEALTH);

// If the NPC has Search go into Search Mode
      //SetSpawnInCondition(NW_FLAG_SEARCH);

// This will set the NPC to give a warning to non-enemies before attacking
      //SetSpawnInCondition(NW_FLAG_SET_WARNINGS);

//Creatures that spawn in during the night will be asleep.
      //SetSpawnInCondition(NW_FLAG_SLEEP);

//This plays Ambient Animations until the NPC sees an enemy or their action list is
cleared.
//NOTE that these animations play automatically for Encounter Creatures.
      //SetSpawnInCondition(NW_FLAG_DAY_NIGHT_POSTING);
      //SetSpawnInCondition(NW_FLAG_APPEAR_SPAWN_IN_ANIMATION);
      //SetSpawnInCondition(NW_FLAG_IMMOBILE_AMBIENT_ANIMATIONS);
      //SetSpawnInCondition(NW_FLAG_AMBIENT_ANIMATIONS);
```

These are self-explanatory for the most part. However, there are a few important notes:

- The special conversations only appear when the PC is perceived by the creature. One is for when the creature is in combat, the other is outside of combat. These are used to give voice to bosses, major allies, and henchmen.
- The **NW_FLAG_SET_WARNINGS** is never set for generic NPCs, monsters, or henchman.
- An uncommon flag, set only in the default **OnSpawn** script for a boss is **NW_FLAG_FAST_BUFF_ENEMY**. This allows enemy bosses to quickly (actually, quite illegally) pump up their defenses before the PC perceives them.
- Another uncommon flag is **NW_FLAG_BEHAVIOR_SPECIAL**. This flag is never set by default but can override the default combat behavior for Neutrally aligned creatures. Currently, it is set up for herbivores, carnivores and omnivores. However, in the default behaviors for most NPCs, it is never used.

Escape Commands

This next section sets various escape behaviors for the Creature object. If set, they cause the creature to flee from an attack or damage in various ways. Only one of the following escape commands should ever be activated at any one time.

OnSpawn Script Escape Commands

```
// OPTIONAL BEHAVIOR (Flee to a way point and return a short time later.)
//SetSpawnInCondition(NW_FLAG_ESCAPE_RETURN);

// OPTIONAL BEHAVIOR (Flee to a way point and do not return.)
//SetSpawnInCondition(NW_FLAG_ESCAPE_LEAVE);

// OPTIONAL BEHAVIOR (Teleport to safety and do not return.)
//SetSpawnInCondition(NW_FLAG_TELEPORT_LEAVE);

// OPTIONAL BEHAVIOR (Teleport to safety and return a short time later.)
//SetSpawnInCondition(NW_FLAG_TELEPORT_RETURN);
```

These are mainly used in the combat-related scripts, such as **OnPhysicalAttacked** and **OnDamaged**.

Custom User-Defined Events

This last section allows the user to fire certain custom events for the Creature object. This mechanism lets you customize the default behavior (by adding specialized code to a user-defined script) while keeping the standard behaviors written by BioWare for the specific event. All your custom code can be stored in the **OnUserDefined** event handler (commonly called **nw_d2_defaultd**) of the creature firing the events. This script for the **OnUserDefined** event handler is meant to be developed by the user and customized as they see fit. Of course, you are always free to write your scripts for other event handlers, but from an organizational standpoint it is very convenient to have it all in one place. The section is as follows:

OnSpawn Script User-Defined Events

```
//OPTIONAL BEHAVIOR - Fire User Defined Event 1001
//SetSpawnInCondition(NW_FLAG_HEARTBEAT_EVENT);

//OPTIONAL BEHAVIOR - Fire User Defined Event 1002
//SetSpawnInCondition(NW_FLAG_PERCIEVE_EVENT);

//OPTIONAL BEHAVIOR - Fire User Defined Event 1005
//SetSpawnInCondition(NW_FLAG_ATTACK_EVENT);

//OPTIONAL BEHAVIOR - Fire User Defined Event 1006
//SetSpawnInCondition(NW_FLAG_DAMAGED_EVENT);
```

```
//OPTIONAL BEHAVIOR - Fire User Defined Event 1008
//SetSpawnInCondition(NW_FLAG_DISTURBED_EVENT);

//OPTIONAL BEHAVIOR - Fire User Defined Event 1003
//SetSpawnInCondition(NW_FLAG_END_COMBAT_ROUND_EVENT);

//OPTIONAL BEHAVIOR - Fire User Defined Event 1004
//SetSpawnInCondition(NW_FLAG_ON_DIALOGUE_EVENT);

//OPTIONAL BEHAVIOR - Fire User Defined Event 1007
//SetSpawnInCondition(NW_FLAG_DEATH_EVENT);
```

In every default script, the related user-defined event is fired. For example, in the **OnPerception** default script, the custom event 1002 is fired. For more information on the other side of this method, see the OnUserDefined Script section later in the chapter.

The OnHeartbeat Script

This event handler is no different than the **OnHeartbeat** handler of a crate. It fires every 6 seconds but in the case of a creature, it can provide crucial and interesting common actions for the creature.

The first few actions are pretty straightforward. If the **NW_FLAG_FAST_BUFF_ENEMY** condition was set in the **OnSpawn** script of the creature (usually the case for enemy boss types), then the creature attempts to set up protections and defenses before the enemy (the heroic PC) can perceive them.

The second action determines if it is currently day or night, and then walks any related set of waypoints (usually a guard path or possibly a random path through a city in the daytime).

Now the real meat of the script comes into play. If the character is sleeping because they spawned in at night and the **NW_FLAG_SLEEPING_AT_NIGHT** spawn condition was set, then a sleeping animation (consisting of Z's floating above their head) is randomly displayed and the script exits.

However, if the creature is not sleeping it is then checked to see if it is currently fighting with an enemy. If so, then **DetermineCombatRound( )** is called and the script exits.

DetermineCombatRound( ) determines the actions the creature will take in the following combat round. As you can imagine it is extremely long and detailed. If the creature is not fighting, it is then checked to see if it has special behaviors (this is rarely true for common creatures and is a special case issue) or is in conversation. If either is true, **DetermineSpecialBehavior( )** is called (for a complete listing of this function, see include file **nw_i0_generic**, under the All Resources section of the Script Editor) and the heartbeat event is signaled if the **NW_FLAG_HEARTBEAT_EVENT** spawn condition is set.

However, if neither condition is true and the **NW_FLAG_AMBIENT_ANIMATIONS** or **NW_FLAG_AMBIENT_ANIMATIONS_AVIAN** spawn conditions were specified, the creature's mobile animations are played. The same action takes place if the creature is spawned in as part of an encounter and there is no enemy present. In addition, if the creature is meant to be immobile (specified by the **NW_FLAG_IMMOBILE_AMBIENT_ANIMATIONS** spawn condition) and there is no enemy nearby, immobile animations are played.

For a closer look at this script, see the **nw_c2_default1** script, located in the All Resources section of the Script browser.

The OnPerception Script

This handler is called when the creature perceives a creature (such as another creature, or the PC). This is a very important script because it allows the creature to react to the environment on a real-time basis.

The default script first checks to see if the creature has the **NW_FLAG_SPECIAL_COMBAT_CONVERSATION** spawn condition set. This condition allows the creature to say a special greeting upon perceiving the player, even during combat. The greeting is designated by attaching the **nw_d2_gen_combat** script to the desired greeting in the "Text Appears When..." tab of the appropriate line in the dialog file. As the creature is actually saying this to himself, don't attach any player responses to the greeting.

Once the greeting part is taken care of, the perception trigger is handled if the creature is not currently in combat. If the last creature was seen

and it is an enemy, the creature faces it and shouts **NW_I_WAS_ATTACKED**. This allows "listening" friends to aid in combat if possible. Combat is then entered, using the ubiquitous **DetermineCombatRound( )**. However, if the last creature was the PC and the creature has **NW_FLAG_SPECIAL_CONVERSATION** spawn condition set, conversation between the two ensues. The conversation bit is designated by attaching the **nw_d2_gen_check** script to the desired conversation node. As the creature is actually saying this to himself, don't attach any player responses to the greeting.

If the last perception was hearing based or if the enemy vanished the creature goes into a search mode where it moves to the point where the last perception occurred and if the enemy is reacquired, determines the actions for the next combat round.

Finally, if the **NW_FLAG_PERCEIVE_EVENT** condition was set, this event is signaled to the creature's **OnUserDefined** script. This allows additional customization of the behavior.

The OnSpellCastAt Script

This handler is called when the creature is the target of a spell in the game. The script is shown here.

The Default OnSpellCastAt Script

```
#include "NW_I0_GENERIC"


void main()
{
     object oCaster = GetLastSpellCaster();
     if(GetLastSpellHarmful())
     {
          if(
           !GetIsObjectValid(GetAttackTarget()) &&
           !GetIsObjectValid(GetAttemptedSpellTarget()) &&
           !GetIsObjectValid(GetAttemptedAttackTarget()) &&
           GetIsObjectValid(GetNearestCreature(CREATURE_TYPE_REPUTATION,
REPUTATION_TYPE_ENEMY, OBJECT_SELF, 1, CREATURE_TYPE_PERCEPTION, PERCEPTION_SEEN))
          )
          {
               if(GetBehaviorState(NW_FLAG_BEHAVIOR_SPECIAL))
               {
                    DetermineSpecialBehavior(oCaster);
               }
               else
               {
                    DetermineCombatRound(oCaster);
               }
               //Shout Attack my target, only works with the On Spawn In setup
               SpeakString("NW_ATTACK_MY_TARGET", TALKVOLUME_SILENT_TALK);
               //Shout that I was attacked
               SpeakString("NW_I_WAS_ATTACKED", TALKVOLUME_SILENT_TALK);
          }
     }
}
```

The script first retrieves the caster of the spell triggering the event. If the spell is considered harmful, the script continues. Then a series of conditions are checked:

- Does the creature currently have a valid attacker?
- Was the creature a valid target for the last spell cast or last attack attempted?
- Was the nearest valid creature (last seen) an enemy of the creature?

If any one check fails the script ends and exits. This indicates that the creature is already in combat or that the last perceived creature was not an enemy. However, if they are all true it is safe to assume that the creature is under attack. Therefore, the creature is checked for any special behaviors that might override standard behavior. If none exists, the behavior for the current combat round is chosen and implemented (using a call to **DetermineCombatRound( )**). Finally, a couple of shouts are fired off, asking for help from other creatures of the same faction and declaring that the creature was attacked. For more information on these shouts, see The OnSpawn Script section.

The OnPhysicalAttacked Script

This handler is called when the creature is physically attacked. Like most combat-related scripts, the script does little if the creature is already in combat. However, let's go through the script in detail.

Three conditions are checked before proceeding to the main part:

- Does the creature have a flee option enabled? If so, the creature flees using whatever option was set in the **OnSpawn** script.
- Is the **NW_FLAG_SET_WARNINGS** spawn condition set? This causes the creature to warn potential targets before attacking. this is rarely set, so we'll assume that it is not present.
- Is the creature currently engaged in combat? If so, the current trigger for this script is ignored and combat continues for the creature.

Let's assume the common case: the creature can't flee, does not warn before attacking and is not in combat. The first action is to validate the last attacker of the creature. If valid, a special behavior check is made. Since this is not set by default we'll assume that the script proceeds normally. If the attacker is still in the immediate area, the creature summons help from possible allies and then determines combat actions for the next round. Once combat is determined, two shouts (stating that the creature was attacked and to attack my target) are fired.

Finally, if the **NW_FLAG_ATTACK_EVENT** condition was set, this event is signaled to the creatures **OnUserDefined** script. This allows additional customization of the behavior.

The OnDamaged Script

This handler is called when the creature takes physical damage. It is very similar in style to the default script for the **OnPhysicalAttacked** event handler but differs slightly.

Three conditions are checked before proceeding to the main part:

- Does the creature have a flee option enabled? If so, the creature flees using whatever option was set in the **OnSpawn** script.
- Is the **NW_FLAG_SET_WARNINGS** spawn condition set? This causes the creature to warn non-enemies before attacking. this is rarely set, so we'll assume that it is not present.
- Is the creature currently engaged in combat? If so, the current trigger for this script is ignored and combat continues for the creature.

Assuming that **FALSE** is returned for all three checks, one of two things occurs:

- If the last attacker is still valid, the creature continues the attack. However, if the creature was surprised, then the creature closes with the new attacker and engages in combat.
- If the creature is attacked by someone other than the current enemy, that new attacker is an enemy and valid, and the new enemy caused damage in excess of 25% of the total HP for the creature or is bigger than the current enemy, that new attacker is the new target for the creature. This causes the creature to focus on the strongest of the current enemies that are attacking it.

Finally, if the **NW_FLAG_DAMAGED_EVENT** condition was set, this event is signaled to the creatures **OnUserDefined** script. This allows additional customization of the behavior.

The OnDisturbed Script

This handler is called when a pickpocket attempt is made on the creature by another creature or a PC. The script is shown on the next page.

The Default OnDisturbed Script

```
#include "NW_I0_GENERIC"


void main()

{

    object oTarget = GetLastDisturbed();


    if(!GetIsObjectValid(GetAttemptedAttackTarget()) &&

!GetIsObjectValid(GetAttemptedSpellTarget()))

    {

        if(GetIsObjectValid(oTarget))

        {

            SpeakString("Thief");

            DetermineCombatRound(oTarget);

        }

        else

        {

            SpeakString("No one stealing");

        }

    }

    if(GetSpawnInCondition(NW_FLAG_DISTURBED_EVENT))

    {

        SignalEvent(OBJECT_SELF, EventUserDefined(1008));

    }

}
```

The script first retrieves the object that last disturbed the inventory of the creature. The owner of the script is then checked to see if they are currently in combat. If not, the object that attempted the pick-pocket is checked for validity. If valid, a thief is discovered and combat is initiated. If the target is not valid (perhaps the thief object was destroyed during processing or teleported away) nothing is done. In either case, an appropriate comment is displayed above the creatures head. Finally, if the **NW_FLAG_DISTURBED_EVENT** condition was set, this event is signaled to the creatures **OnUserDefined** script. This allows customization of the behavior for certain events.

The OnCombatRoundEnd Script

This handler is called at the end of every combat round for the creature. The script is shown on the next page.

The Default OnCombatRoundEnd Script

```
#include "NW_I0_GENERIC"

void main()

{

     if(GetBehaviorState(NW_FLAG_BEHAVIOR_SPECIAL))

     {

          DetermineSpecialBehavior();

     }

     else if(!GetSpawnInCondition(NW_FLAG_SET_WARNINGS))

     {

        DetermineCombatRound();

     }

     if(GetSpawnInCondition(NW_FLAG_END_COMBAT_ROUND_EVENT))

     {

          SignalEvent(OBJECT_SELF, EventUserDefined(1003));

     }

}
```

This script is fairly simple. If the creature has a special behavior state (normal behavior is overridden in certain situations depending on the creature) it is determined. In some cases, this allows the creature to flee or continue attacking. If the **NW_FLAG_SET_WARNINGS** condition was not set (causing the creature to give a warning to non-enemies before attacking), the next combat round begins. Finally, this event occurrence is signaled to the **OnUserDefined** script of this creature object.

The OnConversation Script

This handler is called after dialogue with another creature or a shout is heard. The script is shown here.

The Default OnConversation Script

```
#include "NW_I0_GENERIC"


void main()

{

     int nMatch = GetListenPatternNumber();

     object oShouter = GetLastSpeaker();

     object oIntruder;

     if (nMatch == -1 && GetCommandable(OBJECT_SELF))

     {
```

```
        ClearAllActions();
        BeginConversation();
    }
    else if(GetIsObjectValid(oShouter) && !GetIsPC(oShouter) && GetIsFriend(oShouter))
    {
        if(nMatch == 4)
        {
            oIntruder = GetLocalObject(oShouter, "NW_BLOCKER_INTRUDER");
        }
        else if (nMatch == 5)
        {
            oIntruder = GetLastAttacker(oShouter);
            if(!GetIsObjectValid(oIntruder))
            {
                oIntruder = GetAttemptedAttackTarget();
                SpeakString(GetTag(oIntruder));
                if(!GetIsObjectValid(oIntruder))
                {
                    oIntruder = GetAttemptedSpellTarget();
                    if(!GetIsObjectValid(oIntruder))
                    {
                        oIntruder = OBJECT_INVALID;
                    }
                }
            }
        }
        RespondToShout(oShouter, nMatch, oIntruder);
    }
    if(GetSpawnInCondition(NW_FLAG_ON_DIALOGUE_EVENT))
    {
        SignalEvent(OBJECT_SELF, EventUserDefined(1004));
    }
}
```

First, the script retrieves the listening pattern value and the last shouter. Some examples of common listening patterns are:

Listening Patterns (from NW_I0_GENERIC)

```
//listens for allies who are being attacked
//
SetListenPattern(OBJECT_SELF, "NW_I_WAS_ATTACKED", 1);


//listens for allies who have just died
//
SetListenPattern(OBJECT_SELF, "NW_I_AM_DEAD", 3);
```

For a complete list of the preset patterns, open the **nw_i0_generic** script and search for **SetListeningPatterns**.

If the value is –1 and the action stack of the creature can be modified, the creature enters into conversation (presumably with the PC). However, if the shouter is valid, not a PC object, and a friend, action is taken depending on the value of the shout.

If the value is 4, the placeable object with "NW_BLOCKER" + Blocker creature" tag is retrieved. If the value is 5, the last attacker of the shouter is checked for validity. If not valid, the attempted attack target of the shouter is retrieved. If that is not valid, the attempted spell target of the shouter is retrieved. If that is not valid, determination of the attacker is dropped. In all cases, after the attacker of the shouter has been retrieved, the NPC moves to attack.

Finally, if the **NW_FLAG_ON_DIALOGUE_EVENT** condition was set, this event is signaled to the creature's **OnUserDefined** script. This allows customization of the behavior for certain events.

The OnRested Script

Currently, this event handler has no default script. However, it can be used for customizing the behavior of an NPC attempting to rest (such as a henchman or animal companion following their master's lead), or entering or leaving a current rest cycle.

The OnDeath Script

This handler is called whenever the creature dies from any cause. The script is shown here.

The Default OnDeath Script

```
#include "NW_I0_GENERIC"


void main()
{
    int nClass = GetLevelByClass(CLASS_TYPE_COMMONER);
    int nAlign = GetAlignmentGoodEvil(OBJECT_SELF);
    if(nClass > 0 && (nAlign == ALIGNMENT_GOOD || nAlign == ALIGNMENT_NEUTRAL))
    {
        object oKiller = GetLastKiller();
        AdjustAlignment(oKiller, ALIGNMENT_EVIL, 5);
```

```
    }


    SpeakString("NW_I_AM_DEAD", TALKVOLUME_SILENT_TALK);
    //Shout Attack my target, only works with the On Spawn In setup
    SpeakString("NW_ATTACK_MY_TARGET", TALKVOLUME_SILENT_TALK);
    if(GetSpawnInCondition(NW_FLAG_DEATH_EVENT))
    {
        SignalEvent(OBJECT_SELF, EventUserDefined(1007));
    }
}
```

It is similar to the **OnSpellCast** default script because it identifies the attacker and cries for help. However, before that occurs the alignment of the killer is possibly affected by the NPCs death. If the NPC was a commoner of a level higher than 1 and the alignment was either good or neutral, then the alignment of the killer shifts towards evil by 5. Basically, the more commoner NPCs killed by the killer the more evil he or she becomes. If the **NW_FLAG_DEATH_EVENT** condition is set, this event is signaled to the NPCs **OnUserDefined** script. This allows customization of the behavior or reaction for certain events.

The OnBlocked Script

This handler is called whenever the creature is currently blocked by a door. The script is shown here.

The Default OnBlocked Script

```
void main()
{
    object oDoor = GetBlockingDoor();

    if(GetAbilityScore(OBJECT_SELF, ABILITY_INTELLIGENCE) >= 5)
    {
        if(GetIsDoorActionPossible(oDoor, DOOR_ACTION_OPEN) &&
GetAbilityScore(OBJECT_SELF, ABILITY_INTELLIGENCE) >= 7 )
        {
            DoDoorAction(oDoor, DOOR_ACTION_OPEN);
        }
        else if(GetIsDoorActionPossible(oDoor, DOOR_ACTION_BASH))
        {
            DoDoorAction(oDoor, DOOR_ACTION_BASH);
        }
    }
}
```

This is an interesting script because the action taken is based on the intelligence of the script's owner. The blocking door is first retrieved and if the creature has an Intelligence greater than 5, action can possibly be taken. The intelligence is checked again and if the door can be opened and the Intelligence of the creature is 7 or greater, the door is opened and the creature continues on. If the check fails for any reason the door is checked for the possibility of bashing. If possible, the creature is instructed to bash the door. This script makes for very persistent creatures but keeps any animal intelligence creatures out; like an Ancient Dire Bear.

The OnUserDefined Script

For default creature objects, this event handler is never scripted. However, it can be one of the most heavily customized areas of the creature object. If your creature only has a few unique behaviors, then using a OnUserDefined script and the spawn condition triggers is the best bang for the buck. There is nothing that says you must use this method. If you want to completely rewrite the event handlers for your creature objects, go right ahead. BioWare provided this method as a quick way to modify and tweak default behaviors.

Depending on which spawn conditions were set in the **OnSpawn** script, various custom events are fired at the **OnUserDefined** script of the creature object. By default, these signals all arrive at this script. You can use this script to tweak the default behavior provided by BioWare by adding special behaviors and reactions for the creature to certain events occurring in the game world.

As an example, here is an interesting one taken from a guard dog in the city core. The in-game behavior of these guys is to approach any PC entering "their" territory and bark at them (signaled by the universal "Woof! Woof!" Exclamation). This behavior is triggered by a nearby trigger that defines the guard dog's territory. Here is the **OnUserDefined** script of the guard dog:

Example OnUserDefined Script

```
#include "NW_I0_GENERIC"

void main()
{
    object oIntruder = GetLocalObject(OBJECT_SELF,"NW_L_Intruder");
    object oHome = GetNearestObjectByTag("WP_GUARDDOG");

    if (GetIsObjectValid(oIntruder) &&
        GetDistanceToObject(oIntruder) > 3.0)
    {
        ClearAllActions();
        ActionMoveToObject(oIntruder,TRUE);
        ActionDoCommand(SpeakString("Woof Woof!!!"));
        ActionDoCommand(PlayAnimation(ANIMATION_FIREFORGET_TAUNT));
    }
    else
    {
        ClearAllActions();
        ActionMoveToObject(oHome);
        ActionWait(1.0);
        ActionDoCommand(SetFacing(GetFacing(oHome)));
    }
}
```

The intruder of the space is first retrieved and validated. If the object is far away, the dog runs up to the intruder and barks. Otherwise, the intruder leaves (the only other time this event is triggered) and all actions are cleared from the dog and it returns to the default guard position. It creates a nice urban effect for a PC walking through a town or city.

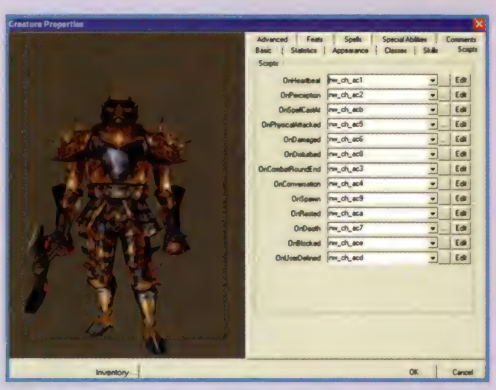
4 Scripting for Henchmen

In general, default scripting for henchman is similar to scripting for creatures. The major differences arise in combat related scripts, the life cycle of the henchman, and the interaction between the henchman and his or her master. What makes henchmen unique is that their scripts are focused towards one thing: aiding and assisting their master. As mentioned before, the default henchman scripts are named using **nw_ch_ac***x*—a set of scripts for all 13 event handlers for the available henchman in the *NWN* campaign, where *x* is 1–9 or a–e (except c).

Making Your Own Henchmen

There's a ton of stuff to consider when making henchmen, and making henchmen from scratch is a process for experienced scripters only. The rest of us can resort to plagiarism; use the basic **nw_ch_ac***x* scripts for all aspects of your henchmen, and make minor modifications if necessary.

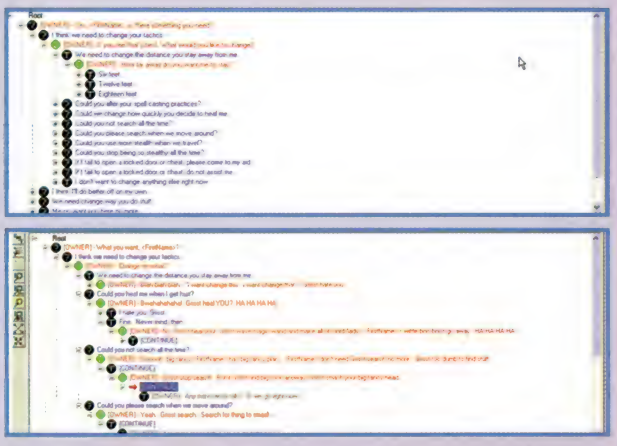
But that's only half the job. The conversation files where you hire, fire, and change the behavior of your henchmen contain dozens of specialized scripts. If you want your henchmen to work basically as they do in the official campaign, then it's easiest to steal the conversation file from a henchman in the game, replace the dialogue line by line (taking care to leave in all the relevant scripts), and delete the stuff that doesn't pertain to your henchman (like the henchmen subplots and requests to use abilities your henchman won't have). If you'll be using the scripts from the Trade of Blades henchmen, you may need to modify the conversation scripts to eliminate references to certain events and chapters.



The basic henchman script set, assuming they haven't changed the names before the game's release!

Pavel from the prologue is much easier to modify, since he has a much tighter dialog file. But Pavel has a few limitations: He can't heal, can't level up, and can't use ranged weapons, so those behavioral hooks aren't in his conversation file (oddly, the spell-casting options are). You can add these lines to the conversation yourself and copy in the appropriate scripts from a different henchman.

You may have noticed that when henchmen level up, they actually destroy themselves and are replaced with a higher level version. Since henchmen can't level up the way PCs do, you'll need to make a version of them for each level, with their blueprint ResRef names incrementally numbered in the same way as the henchmen in the official campaign, so they can be accessed by the **CopyLocals** level-up function used in the **nw_i0_henchman** include file. We haven't yet made any henchman that have the ability to level up, so I must admit we're haven't tested this… But it *should* work. Hopefully we'll have more details about working with henchman in the comment files of our module.



With a bit of creative rewriting, Pavel becomes Grisst. Hey, it's a lot easier than figuring this stuff out from scratch.

OnHeartbeat Scripting for Henchmen

Unlike a generic creature, a henchman has a specific job: Support and aid the master. This is immediately obvious when examining the **OnHeartbeat** script. The actual script (**nw_ch_ac1**) is quite involved so we will break it up into logical sections. We don't have the space to print it here, so you'll have to pull it up on your PC if you want to follow along.

Because the script is a series of nested **if-then-else** statements (one inside of another), there are checks that occur before action is taken. We will look at the script by focusing on each **if-then-else** statement.

First off, if the henchman is currently busy (fighting, disabling a trap, and so on) the script "falls out" the end (or finishes execution).

However, if the henchman is currently available, the henchman is initialized for trap disarming. If the following conditions are all true, the henchman… (Actually, because I am getting tired of typing "the henchman" let's just call him George. Anytime you see me reference George, I am talking about the henchman). Anyway, if the following conditions are all true, George attempts to disable the trap (and then the script exits):

- George has a valid master.
- George has at least one skill rank in Disable Traps and, obviously, has the skill Disable Traps.
- George is not in conversation.
- The trap is a valid trap object.
- George is currently **not** disabling a trap.
- George has at least one level in the Rogue class.

However, if the previous check failed, the next check determines if George is currently performing any of the following actions: following, disabling traps, opening locks, or attacking. If any check returns true, then George is "busy" and the script exits, after firing the custom 1001 event if the **NW_FLAG_HEARTBEAT_EVENT** condition was set in George's **OnSpawn** script.

If they all return false, the script proceeds with another check. This involves the following conditions for George:

- Is there a valid attack target?
- Is there a valid attempted spell target?
- Is there a valid attempted attack target?
- Is there an enemy creature nearby?

Basically, we are checking to see if he is in combat or not. If any or all return true, he is in combat and can't be bothered. Therefore, George sets up for the next round of combat (**DetermineCombatRound( )**), fires the custom 1001 event (which you can add to the OnUserDefined script) and exits. However, if all conditions are false, he is not in combat and the script proceeds.

The next section is a series of rapid checks that narrow down his current state. Basically, they determine if George is currently farther away from his master than the current following distance. If so, all actions are cleared and George closes the distance between him and his master. If able to, George moves silently and searches while catching up.

However, if he is within the agreed-upon follow distance to his master and George is currently not standing his ground, the script proceeds. If the master is not currently resting, George follows his movements employing his Move Silently and Search skills if available. The script then exits, firing the custom 1001 event.

If George was not in combat but there was an enemy nearby, he enters combat with the nearest enemy. The last script exit occurs, and the custom 1001 event fires, if it has been placed in the henchman's **OnUserDefined** script.

OnPerception Scripting for Henchmen

Unlike the **OnHeartbeat** script, this script only differs slightly from the default creature **OnPerception** script. The henchman version has two possible actions.

If the henchman (or George) is not standing his ground and not in combat, the last creature seen is checked. If it is an enemy, the henchman engages it in combat. If it is the PC, a special conversation (exactly like the creature special and combat conversations) is initiated.

The second possibility is that George is already in combat and can safely ignore whatever was last perceived. In this case, the custom 1002 event is fired and the script exits.

OnPhysicalAttacked Scripting for Henchmen

In this script, the primary motive is to protect the master, if present. If alone, the henchman attacks the object causing the damage.

The default script first checks to see if the henchman (or George) is busy. If not and he is not standing his ground, the script proceeds. If George is currently not in combat and was actively defending his master, he continues fighting his master's attacker. Otherwise, he initiates combat with the new attacker.

The final action of the script is to fire the custom 1005 event, if applicable.

OnDamaged Scripting for Henchmen

In this script, if George is in combat, the script exits automatically. If he is standing his ground or following the script also exits but first fires the custom 1006 event, if applicable.

However, if George has no valid attackers or spell targets then he attacks whomever is attacking his master.

If George already has a valid attacker, it is checked to see if it delivered the damage that caused the script to trigger. If it is, the attack continues. If it isn't, and it's an enemy that took more than 25% of George's maximum HP in the last attack, George focuses on the new, more powerful threat.

The final action of the script is to fire the custom 1005 event.

OnConversation Scripting for Henchmen

This script responds to an attempt to start conversation by its master or the shouting of its master.

In the first case, if the object wishes to initiate a conversation with George, all actions are cleared and conversation begins.

However, if someone shouted (for example **NW_I_WAS_ATTACKED**) and it was George's master, then George rushes to the rescue. Finally, the script fires the custom 1004 event, if applicable.

OnSpawn Scripting for Henchmen

Serving the same purpose as the default **OnSpawn** script for creatures, this version (**nw_ch_ac9**) first initializes the henchman states and sets up special listening patterns for the henchman.

Example Henchman OnSpawn Script

```
// Initialize the standard commands for which Associates need to be listening for.
    SetAssociateListenPatterns();

// Goes through and sets up which shouts the henchman will listen to.
    SetListeningPatterns();


    SetAssociateState(NW_ASC_POWER_CASTING);
    SetAssociateState(NW_ASC_HEAL_AT_50);
    SetAssociateState(NW_ASC_DISTANCE_2_METERS);
    SetAssociateState(NW_ASC_RETRY_OPEN_LOCKS);
    SetAssociateState(NW_ASC_DISARM_TRAPS);
    SetAssociateState(NW_ASC_MODE_DEFEND_MASTER, FALSE);
    SetAssociateState(NW_ASC_USE_RANGED_WEAPON, FALSE); //User ranged weapons by default
if true.


    SetAssociateStartLocation();
```

By default, the following states are set:

- Doubles the CR of any spell cast by the henchman. This makes it harder for the target to resist any spell effects.
- If master is below 50% of maximum HP, heal the master.
- The distance the henchman keeps between him and the master.
- Attempts to open locks if master failed.
- Disarms any nearby traps.
- Does not defend master.
- Use melee weapons primarily in combat.

After this section, as with the creature script, a special conversation and special combat conversation can also be initialized (using the same method as the creature). For more information, see the "OnSpawn Script" section.

The remaining script code is commented out but can be used to set up various custom events to fire at appropriate times (such as **OnPerceive**, **OnDeath**, and so forth). However, unlike the creature script, the henchman script does not generate any treasure or make the henchman walk waypoints.

OnDeath Scripting for Henchmen

The **OnDeath** henchman script is very similar to the default **OnPlayerRespawn** script.
First, all harmful effects are removed, then the henchman is resurrected and all hit points are restored. After restoration, the henchman is returned to the Hall of Justice near the Recall Stone portal to wait for the arrival of the PC. You'll need to change the **JumpToObject** destination and such to make your henchman respawn in a way that's relevant to your campaign (if you make no changes, they'll simply respawn at the point where you died).

This same script is apparently used for a mage's familiar, and in that case the script applies 1d6 damage to the mage.

3 Custom Behavior Scripting for Creatures

If all your NPCs did was use the default scripting, they would be a responsive and emotive lot: some wandering aimlessly, others in conversation or arguing a point, or defending themselves against enemies or attackers. However, BioWare never intended you to stop there. There is a whole world of customized behavior that awaits you. In this section we'll look at some basic modifications that breathe life into the module and your NPCs.

Talking the Talk

There are a host of animations for NPCs that do different things, beyond the default animations for immobile NPCs. These are enabled by uncommenting a line in the default script (**nw_c2_default_9**) that sets the **NW_FLAG_IMMOBILE_AMBIENT_ANIMATIONS** flag.

There are two basic types: fire and forget (the animation happens once) and looping (the animation continues until the actions of the NPC are cleared, the NPC is attacked, or the animation time limit expires).

Some common fire and forget animations are listed here:

- ANIMATION_FIREFORGET_DRINK
- ANIMATION_FIREFORGET_PAUSE_SCRATCH_HEAD
- ANIMATION_FIREFORGET_PAUSE_BORED
- ANIMATION_FIREFORGET_TAUNT
- ANIMATION_FIREFORGET_VICTORY1

Their purpose is self-explanatory. For a full list, check out the Constants list (beginning with **ANIMATION**) in the Script Editor.

Here is a list of common looping animations:

- ANIMATION_LOOPING_GET_LOW
- ANIMATION_LOOPING_LOOK_FAR
- ANIMATION_LOOPING_MEDITATE
- ANIMATION_LOOPING_SIT_CHAIR
- ANIMATION_LOOPING_TALK_LAUGHING

These can be initiated at any time. Some common methods are looping animations in the **OnSpawn** script and fire and forget animations at unique or random times. For an example of plugging in a looping animation, see Chapter 1 and Appendix C (Scripts Deciphered section).

Example: Welcome to the Crawl 'n' Puke

As an example of NPC scripting, we're going to set up a tavern called the Crawl 'n PUke and a realistic crowd of patrons. We'll start by setting up a few animations (triggered by **OnHeartbeat** events) and settle some people down at a table. Here are a two samples of **OnUserDefined** scripts that demonstrate this:

OnUserDefined Script

```
void main()

{

   int nUser = GetUserDefinedEventNumber();
```

```
  switch(nUser)
  {
    case 1001: //OnHrtbt
    if(!IsInConversation(OBJECT_SELF))
      ActionPlayAnimation(ANIMATION_LOOPING_TALK_LAUGHING, 1.0, 2.0);
      ActionWait(1.0);
      ActionPlayAnimation(ANIMATION_FIREFORGET_DRINK);
    break;
  }
}

void main()
{
  int nUser = GetUserDefinedEventNumber();


  switch(nUser)
  {
    case 1001: //OnHrtbt
    if(!IsInConversation(OBJECT_SELF))
      ActionPlayAnimation(ANIMATION_LOOPING_TALK_PLEADING, 1.0, 3.0);
    break;
  }
}
```

Here is a different type of OnUserDefined script that sits a dwarf down after a tiring campaign:

OnUserDefined Script

```
void main()
{
  int nChair = 1;
  int bFoundChair = FALSE;
  object oChair;
```

```
int nUser = GetUserDefinedEventNumber();



switch(nUser)
{
  case 101:  //OnSpawn event
    oChair = GetNearestObjectByTag("CF_CHAIR", OBJECT_SELF, nChair);
    ClearAllActions();

    while (bFoundChair == FALSE && GetIsObjectValid(oChair) == TRUE)
    {
      if (GetIsObjectValid(GetSittingCreature(oChair)) == FALSE)
      {
        bFoundChair = TRUE;
        ActionSit(oChair);
      }
      else
      {
        nChair++;
        oChair = GetNearestObjectByTag("CF_CHAIR", OBJECT_SELF,nChair);
      }
    }
    if (bFoundChair == FALSE)
      ActionSpeakString("Couldn't find a chair");
    break;
}
```

Basically, the script is fired once (from the **OnSpawn** script), causing Nulir to sit only once, and not every 6 seconds. The script first initializes some variables and gets the custom event number just fired. If it is 101 (our sitting event), the script continues. Inside the **case** statement the nearest chair is grabbed and checked to see if it is occupied. If not, the dwarf is told to sit in it, and **bFound** is set to TRUE. This breaks the loop and eventually exits the script. However, if it is occupied the next chair with the "CF_CHAIR" tag (replace this with the tag of whatever chairs are in your area) is grabbed and the process repeats until Nulir takes a seat or he runs out of chairs.

Walking the Walk

In some cases, an immobile NPC talking to himself just won't cut it. For more mobile needs, such as various townspeople on errands or a guard on patrol duty, there are several methods of providing the proper motivation. There are three basic methods for scripting NPC movement:

- Using pre-existing waypoints and the **WalkWayPoints( )** function.
- Walking to the location of a pre-existing object (including the waypoint object).
- Jumping to a location or object.

The WalkWayPoints( ) Function

This function is called in the default **OnSpawn** script for NPCs. This allows an NPC to walk a series of waypoints in a certain order. The function works by first looking for any waypoints in the current module that have the tag "**WP_" + npc tagname + "_0X**".

For instance, if you had an NPC named George, he would walk, after spawning in, a series of waypoints with the following tags:

```
WP_George_01, WP_George_02, and WP_George_03.
```

You can specify any number of waypoints and place them in any area of the module. There are some interesting parameters for **WalkWayPoints( )**. By default, the NPC will walk his or her waypoints, pausing at each one for a second. However, you can also make the NPC run the waypoint series, pausing for a specified time period (from 0 seconds upward).

However, if the associated waypoint of the NPC has the form of "**POST_" + npc tagname** (for instance, "**POST_George**") then the NPC returns to that waypoint after engaging in any combat. This is useful for keeping a specific guard at his or her post or any NPC in a certain location no matter what. There are numerous examples of this functionality in the City Core area of Chapter 1, in the official *Neverwinter Nights* campaign.

Example: Run Away!

The problem is to create a traumatized caravan member who runs away at the sight of the PC. In this solution, the NPC simply runs directly to a waypoint hidden beside a large cliff. Of course, you can get more extravagant by adding other waypoints or behaviors to the script but the solution remains the same.

The solution is to modify the default **OnSpawn** script slightly (by removing the call to **WalkWayPoints( )**) and to create a simple **OnPerception** script that causes the NPC to run at the sight of the NPC. Here is the **OnPerception** script.

OnPerception Script

```
#include "NW_I0_GENERIC"


void main()

{

 if(GetIsPC(GetLastPerceived())) //run away!

 {

 SpeakString("Ahhh! Bugs... bugs EVERYWHERE!!");

 WalkWayPoints(TRUE, 0.0);

 };

}
```

The **NW_I0_GENERIC** include file is needed for the call to **WalkWayPoints( )** later in the script. The script first checks that the last creature perceived was the PC object. If the check is true, then the NPC screams out a short line of gibberish and runs to his "hiding spot", marked by a waypoint with the tag **WP_Vaster_01**. For the call to **WalkWayPoints**, his **TRUE** parameter forces the NPC to run, and the second parameter is the time he pauses at each waypoint.

Moving to Specific Objects

This method is very useful for performing seemingly random walking actions or allowing the NPC to choose a path to a specific location or object. In a later example, we use this to make our waitress at the Crawl 'n Puke serve drinks to the local customers. The following functions can be used to implement this method:

- **ActionMoveToLocation** – Moves the script owner (or **OBJECT_SELF**) to a specified location. Pathing to this location is automatically handled by the owner and the game engine.
- **ActionMoveToObject** – Moves the script owner (or **OBJECT_SELF**) to a specified object. Pathing to this object is automatically handled by the owner and the game engine.
- **ActionMoveAwayFromLocation** – Moves the script owner (or **OBJECT_SELF**) away from a specified location. You specify how far away the NPC moves.
- **ActionMoveAwayFromObject** - Moves the script owner (or **OBJECT_SELF**) away from a specified object. You specify how far away the NPC moves.
- **ActionRandomWalk** – I just added this because it also makes the NPC move around. This is very nice for "window dressing" a town wth some random NPC behavior. There are no parameters (hence the name **ActionRandomWalk**).

Note that these functions (except for **ActionRandomWalk( )**)allow running or walking for the movement action. In addition, these all add actions to the owner's action stack. This means that they will always occur in the correct sequence but not necessarily at the exact moment they are executed in the script.

Example: Waiting Tables at the Crawl 'n Puke

Now that we have our bar full of patrons, it's time to give it some action. One of the usual fixtures in a bar (at least in the bars with good service) is a waitress that takes and delivers your drinks. We need a waitress at the Crawl 'n Puke that seems to randomly take orders from the customers in the bar.

We'll take a default Waitress and modify her **OnSpawn** and **OnUserDefined** scripts. For her **OnSpawn** script, we start with the default one (**nw_c2_default9**) and modify it by initializing a few local variables and setting the **NW_FLAG_IMMOBILE_AMBIENT_ANIMATIONS** and **NW_FLAG_HEARTBEAT_EVENT** conditions. For the generic behaviors we will comment the **WalkWayPoints** and **GenerateNPCTreasure** functions (we have our own customized method). We compile and save this as a separate script (**cf_wait_def9**) and add it to the **OnSpawn** event handler of the waitress.

Now comes the cool part: the scripting of her doing her job. For this part, we populate the bar with patrons, and plop down a bar and bartender. All the patrons have the same tag; **CF_PATRON**. This will be used to pick random customers in the **OnUserDefined** script (**cf_waitress_defd**). Now for the actual script:

Waitress's OnUserDefined Script

```
void DrinkOrder();

void main()

{

   int nUser = GetUserDefinedEventNumber();


   object oBar = GetWaypointByTag("CF_BAR");

   object oPatron = GetLocalObject(OBJECT_SELF, "CF_PATRON");
```

```
object oBartender = GetObjectByTag("CF_BARTENDER");

switch(nUser)
{
  case 1001: //OnHrtbt
    int nRandom = d6();
    while(nRandom == GetLocalInt(OBJECT_SELF, "CF_LASTCUSTOMER"))
      nRandom= d6();

    if(!GetIsObjectValid(oPatron))
    {
      oPatron = GetNearestCreature(CREATURE_TYPE_PLAYER_CHAR, PLAYER_CHAR_NOT_PC,
            OBJECT_SELF, nRandom);
      if(GetIsObjectValid(oPatron) && oPatron != OBJECT_SELF && oPatron != oBartender)
      {
        //Move to Customer
        SetLocalInt(OBJECT_SELF, "CF_LASTCUSTOMER", nRandom);
        SetLocalInt(OBJECT_SELF, "BARMAID_STATE", 1);
        SetLocalObject(OBJECT_SELF, "CF_PATRON", oPatron);
        ActionMoveToObject(oPatron);
        ActionSpeakString("What's your poison?");
        ActionWait(2.0);
        DrinkOrder();
        //Move to Bar
        ActionDoCommand(SetLocalInt(OBJECT_SELF, "BARMAID_STATE", 2));
        ActionMoveToObject(oBar);
        ActionSpeakString(GetLocalString(OBJECT_SELF, "CF_ORDER"));
        ActionWait(2.0);
        //Move back to the customer
        ActionDoCommand(SetLocalInt(OBJECT_SELF, "BARMAID_STATE", 3));
        ActionMoveToObject(oPatron);
        ActionSpeakString(GetLocalString(OBJECT_SELF, "CF_AMOUNT"));
        ActionWait(1.0);
```

```
            ActionDoCommand(SetLocalObject(OBJECT_SELF, "CF_PATRON", OBJECT_INVALID));
          }
        }
        break;
    }
}
```

The script basically picks a random customer and sends Tara over to get the drink order, back to the bar to pick it up, and finally back to the customer for the delivery and payment. If Tara is currently busy with a customer, the script bails out and does nothing. Because this fires every 6 seconds it's a good bet she is busy most of the time.

Let's break it down with some more detailed discussion. First, some important variables are initialized, containing the user-defined event just fired, a random patron (chosen until different than the last patron served), and her work station.

If the fired event is **OnHeartbeat** and she currently has no patron, we begin her movements. The next customer is grabbed (and the last customer variable updated to prevent repetition) and if it is a valid object and not Tara or the bartender, she sets her current state and the current customer object. She then walks over to the thirsty customer and asks for an order. She then updates her state (a bookkeeping thing not essential to the script) and walks back to the bar (or at least the waypoint defined near the bar). She gives Vicho her order, picks up the drink, updates her state again and walks back to the customer. When she arrives, her state updates and she asks for the payment.

To indicate she is ready for another customer, she sets the local object variable for the customer to **OBJECT_INVALID**.

The final touch (the **DrinkOrder** function) is a bit of randomization for the observing PC. Here is the **DrinkOrder( )** function that randomizes the strings the PC sees while Tara does her job.

Drink Order Function

```
void DrinkOrder()
{
 int nOrder= -1;
 string sOrder;
 string sAmount;

 nOrder= d4();
 switch(nOrder)
 {
  case 1:
    sOrder= "Vicho, one whiskey ditch";
    sAmount= "That'll be a silver.";
    break;
  case 2:
    sOrder= "Gimme a Dragon's Breath.";
```

```
    sAmount= "A copper, if you please.";
    break;
  case 3:
    sOrder= "One Foofy Fairy for the tough guy in the corner.";
    sAmount= "That'll be a gold... my dear.";
    break;
  case 4:
    sOrder= "I need two ales Vicho";
    sAmount= "Two coppers, buddy.";
    break;
 }

 SetLocalString(OBJECT_SELF, "CF_ORDER", sOrder);
 SetLocalString(OBJECT_SELF, "CF_AMOUNT", sAmount);
}
```

The end result is a waitress constantly taking orders and dispensing drinks to thirsty customers. Of course, you can take this much further by providing more random bits of conversation, making the customer actually face the waitress, and so on. That's the beauty of the *NWN* toolset.

Example: Jumping to a Location or Object

After returning from the caves in the first scene of our module, we imagine the surviving caravan NPCs will have reassembled at the caravan site. Part of the solution code is added to an earlier script that deactivated or activated encounters based on a variable. The solution is two parts: a trigger and a small script causing all NPCs to jump to the caravan site.

The trigger code is placed in the encounter script we talked about in Chapter 3. Whenever the encounters are deactivated, this trigger fires.

The Script Trigger

```
object oGrisst;
object oNashina;
object oGeorge;

oGrisst= GetObjectByTag("Grisst");
oNashina= GetObjectByTag("Nashina");
```

```
oGeorge= GetObjectByTag("George");


if(GetIsObjectValid(oGrisst))
  SignalEvent(oGrisst, EventUserDefined(101));
if(GetIsObjectValid(oNashina))
  SignalEvent(oNashina, EventUserDefined(101));
if(GetIsObjectValid(oGeorge))
  SignalEvent(oGeorge, EventUserDefined(101));
```

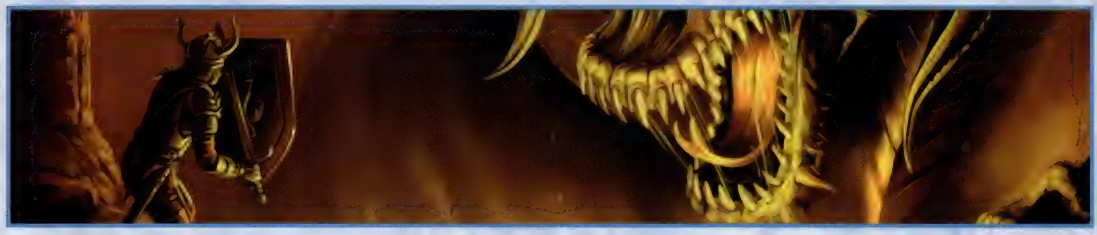
Every caravan NPC object is first retrieved, and then each is validated and "pinged" with a custom event signal.

The script for the **OnUserDefined** handler of each NPC is the same except for the waypoint locations they each jump to.

OnUserDefined Script

```
void main()
{
  int nUser = GetUserDefinedEventNumber();
  location lLoc;
  switch(nUser)
  {
   case 101: //Jump to lake
    lLoc= GetLocation(GetObjectByTag("CF_WP_NASHJUMP"));
    ClearAllActions();
    JumpToLocation(lLoc);
  }
}
```

The script first retrieves the custom event identity just fired. Because this is the only custom event (at this point), it goes directly to the jumping code. The jump location is retrieved by grabbing the location of each NPCs special waypoint (in this case, it is Nashina's). Then all current actions are cleared from the NPC and they are whisked away to the caravan site instantly and magically (at least to them).



Example: The Climactic Battle

Now let's put everything we've learned together and make a scripted event: A climactic battle between a high-level wizard (let's call him Fishbanger Danode'for) and Lord Neven. The PC enters the room, triggering the encounter. Mayhem ensues and just before the end, the PC is spirited away to another location.

We'll need to create three **OnUserDefined** scripts for the three NPCs (Lord Neven, Fishbanger, and Lord Neven's apprentice). The trigger can be of any type and since we don't want to hurt the PC the majority of spells are fake.

Here is Fishbanger's **OnUserDefined** script:

Fishbanger's OnUserDefined Script

```
void main()

{

  int nUser = GetUserDefinedEventNumber();


  switch(nUser)

  {

    case 500: //taunt Lord Neven

      ActionSpeakString("Lord Neven... you suck!");

      ActionDoCommand(SignalEvent(GetObjectByTag("LordNeven"), EventUserDefined(100)));


      break;


    case 200: //LN responded

      ActionMoveToObject(GetObjectByTag("CF_WP_CASTSPELL"));

      ActionCastFakeSpellAtObject(SPELL_FEEBLEMIND, GetObjectByTag("LordNeven"));

      DelayCommand(2.0, SignalEvent(GetObjectByTag("LordNeven"), EventUserDefined(110)));

      ActionCastFakeSpellAtObject(SPELL_FIREBALL, GetObjectByTag("LordNeven"));


      break;


    case 300: //LN casting MeteorStorm

      ActionCastFakeSpellAtObject(SPELL_FINGER_OF_DEATH, GetObjectByTag("LordNeven"));

      break;

  }

}
```

Basically, he has three events. The first event is triggered by the PC entering the room. Fishbanger steps up and taunts Lord Neven. He then signals Lord Neven to reply. You will see the other sides of this event in later scripts.

The second event is after Lord Neven's reply. Fishbanger moves up closer, and then starts flinging some spells at Lord Neven, signaling Lord Neven in the middle of the attack to defend himself.

The final event is triggered by Cara, Lord Neven's assistant as she prepares to teleport the PC away from the battle. Fishbanger fires off a final spell just before the PC is spirited away.

Now let's examine Lord Neven's script.

Neven's OnUserDefined Script

```
void main()
{
  int nUser = GetUserDefinedEventNumber();

  switch(nUser)
  {
    case 100: //respond to taunt
      ActionWait(1.0);
      ActionSpeakString("Oh yeah! Well... you suck harder!!");
      ActionDoCommand(SignalEvent(GetObjectByTag("Fishbanger"), EventUserDefined(200)));

      break;

    case 110: //Fishbanger is attacking
       ActionCastSpellAtObject(SPELL_GREATER_STONESKIN, OBJECT_SELF, METAMAGIC_QUICKEN);
       ActionCastSpellAtObject(SPELL_ELEMENTAL_SHIELD, OBJECT_SELF, METAMAGIC_QUICKEN);
       ActionSpeakString("Cara! Get RID of these fools!!");
       ActionDoCommand(SignalEvent(GetObjectByTag("Cara"), EventUserDefined(210)));

      break;

    case 300: //Cara is getting ready to teleport PC out
      ActionCastFakeSpellAtObject(SPELL_METEOR_SWARM, GetObjectByTag("FishBanger"));
      DelayCommand(1.0, ActionDoCommand(SignalEvent(GetObjectByTag("Cara"),
EventUserDefined(220))));
      break;
  }
}
```

The first event responds to Fishbanger's taunt. Lord Neven then signals the next round of action. In the second event, Lord Neven protects himself against attack in response to Fishbanger's first assault. He then tells Cara to get rid of the PC by any means. In the last event he casts Meteor Swarm, and then signals Cara to teleport away the PC.

Finally, Cara's script:

Cara's OnUserDefined Script

```
void JumpPC(object oPC);

void main()

{

  int nUser = GetUserDefinedEventNumber();

  object oPC;


  switch(nUser)

  {

    case 210: //LN orders the PC out

      ActionSpeakString("As you command.");

      DelayCommand(2.0, ActionDoCommand(SignalEvent(GetObjectByTag("LordNeven"),
EventUserDefined(300))));

      DelayCommand(2.0, ActionDoCommand(SignalEvent(GetObjectByTag("Fishbanger"),
EventUserDefined(300))));


      break;


    case 220: //LN casting Meteor Swarm

      oPC= GetFirstPC();

      ActionCastFakeSpellAtObject(SPELL_POWER_WORD_KILL, oPC);

      DelayCommand(2.0, JumpPC(oPC));
```

```
        break;
    }
}
void JumpPC(object oPC)
{
    object oDest= GetObjectByTag("WP_CF_AT_Room4");
    location lLoc= GetLocation(oDest);

    AssignCommand(oPC, JumpToLocation(lLoc));
}
```

This script is made up of two functions: one handles the custom events and the other teleports the PC to a different room. In the event-handling portion, Cara first acknowledges Lord Neven's command to spirit away the PC. She then signals Lord Neven and Fishbanger to begin their final assaults. In the final event, she grabs the PC object, and acts like she is casting some big spell. In reality, a call is made to a custom function, **JumpPC( )**. This function gets a location in a different area, and then jumps the PC to that location. End of climactic battle sequence.

As you've noticed in these three scripts, there are a lot of timing issues that are handled by using **DelayCommand( )** and modifying the individual action stacks for the NPCs so that everything happens in the exact order required.

Fighting the Good Fight

Another interesting thing creatures do is fight. Usually they seem to have an unhealthy fixation on ending the life of the PC in as short a fashion as possible but beside the rampant and simple bloodlust, you can customize this so that your creatures seem a cut above the rest.

Using the default behavior scripts gets you a lot for your money and with a little tweaking you can create interesting behaviors for your creatures. Some examples include:

- Exclusively targeting specific PCs based on a condition (such as the strongest or the one currently with the most hit points). For instance, you could have a group of two wizards in the back targeting the biggest PC while their guards hold that burly warrior off for a few rounds.
- Setting up some specific taunts or one-liners based on the PC's class or affiliation. Let's say your module has a focus on the Drow sub-race. You could create two creature groups: a group of dwarven mercenary and a Drow war party. When the PC encounters both and engages in battle, there could be a 50% chance that the two groups would start attacking each other and not the PC!
- There is a neat function in *Neverwinter Nights* called **SurrenderToEnemies( )**. This function causes all enemy creatures in the immediate area to stop all actions and become neutral for a period of time. Keeping this function in mind, you could set up a small group of bodyguards and a sorceress. When the sorceress dies, the bodyguards could surrender or perhaps when the sorceress takes combat damage for the first time she could immediately surrender.

Let's look at that last example in more detail.

Example: Scripting Morale Failure

The problem is to create a small group of creatures, with one leader-type creature. Once the leader dies, the remainder of the group should surrender due to a lack of morale.

The solution is to modify the **OnDeath** script of the leader to send a custom event to all party members, and then **OnUserDefined** scripts for the other party members causing them to surrender based on the custom event.

The modified **OnDeath** script for the leader follows on the next page.

OnDeath Script

```
void main()
{
  if(GetLocalInt(OBJECT_SELF, "CF_SURRENDER") == 0)
  {
    SignalEvent(GetObjectByTag("CF_GUARD1"), EventUserDefined(101));
    SignalEvent(GetObjectByTag("CF_GUARD2"), EventUserDefined(101));
    SignalEvent(GetObjectByTag("CF_GUARD3"), EventUserDefined(101));
    SetLocalInt(OBJECT_SELF, "CF_SURRENDER", 1);
  }

  if(GetSpawnInCondition(NW_FLAG_DEATH_EVENT))
  {
    SignalEvent(OBJECT_SELF, EventUserDefined(1007));
  }
}
```

Since we only want the surrender to happen once, we track the surrender event with a local variable. If **CF_SURRENDER** is 0 (leader just died for the first time), a custom event is signaled to the other members of the party. And then the **CF_SURRENDER** variable is set to 1, so this doesn't happen again.
Here is one of the **OnUserDefined** scripts for a party member:

OnUserDefined Script

```
void main()
{
  int nUser = GetUserDefinedEventNumber();

  switch(nUser)
  {
   case 101: //Surrender
```

```
    SurrenderToEnemies();

    ClearAllActions();

    SpeakString("Pleease... no hurt us!!");

    break;

  }


}
```

This is a very simple script that checks for the "surrender" event (101). If the event was fired, the leader has died and it is time to cower in defeat. First a call to **SurrenderToEnemies( )** is made, and then all actions are cleared, and finally each member says something, indicating their surrender.

Of course, you could do other things like cause the surviving members to run away or automatically die. Once again, using the Neverwinter Toolset, the possibilities are plentiful.

4 NPCs and the Conversation Editor

The **Conversation Editor** is an incredibly sophisticated and important tool. Conversations with NPCs add flavor and realism to your world, and are the player's most important means of gaining information about the quest and its setting. But above and beyond that, virtually all storyline progression takes place within dialog files. 90% of the time it's a dialog file that sends the player on a quest, gives the player clues and plot items during the quest, and rewards the player for completing the quest. For this reason, the dialog files of major characters tend to be packed full of scripts. Some characters in the official campaign have over 40 different scripts referenced in their dialog file!

Fortunately, the **Script Wizard** built into the Conversation Editor will happily make nearly all of these scripts for you. And the dialog files allow you to add many things that usually require scripting, like creature animations, sounds, and journal editing, with no scripts at all.

Understanding Dialog Files

The Conversation Editor is the only way to create a **dialog file**, which contains everything a specific character is capable of saying (outside of any "SpeakString" functions that appear, very rarely, in scripts), and the scripts for any actions that result during conversations with that character. For characters who just stand around in town, the Dialog File can be nothing more than a single line of text that pops up over their head. For major characters, it can be up to thousands of lines, organized in a complicated flow chart. For an example of a moderately complex conversation, let's open up the Prologue and take a look at Aribeth's dialog file. Select her character, in the Graduation Chamber, open the Creature Properties window, and hit the "Edit" button under Conversation, bringing up the **m0q01a09arib** dialog file in the Conversation Editor.

One Liner Conversations

Aribeth's conversation nodes

As you can see, Aribeth has six different conversations nodes coming off the Root directory. Three of these (the ones that can't be expanded) are "one liners", nodes that have only one line of text and are displayed over the speaking character's head, instead of in a Conversation Window. Unless a one liner is all a character has, these are not spoken when you click on a character with the Conversation icon. Instead, one liners are called by the **SpeakOneLinerConversation** function in a script. There are no special scripts within the dialog file required to make a one liner appear. If the **SpeakOneLinerConversation** function is called, with Aribeth as the target, Aribeth will automatically say the topmost one liner in her dialog file unless a Starting Conditional script prevents it. If a Starting Conditional script does prevent it, it goes to the next one down and repeats the process.

Why Aribeth Says What When

All of the scripts under **"Text Appears When..."** tab are **Starting Conditionals**. These scripts return a 0, meaning "True", or a 1, meaning "False". If it returns a True, the owner of the conversation speaks the line of dialog in this node. If it's False, the program knows that it's the wrong time to speak this line and moves down the list. This works both for Aribeth's dialog and the player's possible responses. For example, half of the player's possible responses in this file will never be shown as options in the conversation, because the Starting Conditional script for those lines will check the player's intelligence and return a False, because the player character is too dumb or too smart.

The Script Wizard can make Starting Conditional scripts for you based on Abilities (most commonly intelligence or charisma), Character Class, Gender, Race, the presence or absense of Feats or Skills, a Skill Check, the presence or absence of an Item in your Inventory, the value of a Local Variable stored anywhere in the module, or even on a randomly generated number. This is because the Script Wizard loves you.

Let's look at the Starting Conditionals of Aribeth's six conversation nodes and see if we can figure out at what times each conversation is spoken. The script for "We are under attack! Destroy the invaders!" reads:

```
int StartingConditional()

{

return !GetIsObjectValid(GetPCSpeaker());

}
```

This statement appears frequently. Basically, this just makes sure that the PC isn't currently in conversation with the speaker. The "I must speak to you" conversation adds another condition, that the Local Variable "NW_PROLOGUE_PLOT" (assigned to the module) be equal to 99. A script elsewhere in the module sets this integer at 99 after the first wave of the attack on the Academy. So during combat, when Aribeth's **SpeakOneLinerConversation** function is first called, she says "We are under attack! Destroy the invaders!" and after combat, when the NW_PROLOGUE_PLOT variable is set, she says "I must speak to you" instead. Since those two one-liners seem to cover all possible scenarios, I can't imagine any reason Aribeth would ever say her final one-liner.

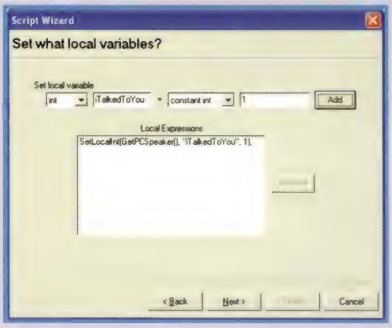
Her three multi-line conversation nodes each have a starting conditional as well. Like the one-liners, they are in reverse chronological order. Putting your conversations in this order is a good idea, because as new variables are set, the conditions for the earliest conversation (which often has no Starting Conditional) may still be met, preventing any node below it from ever being read. I apologize if that last sentence made no sense whatsoever (that's after thirty rewrites, too). Just trust me on this one: The more specific the Starting Conditional, the higher it should go in the dialog file.

Anyway, we'll read them from the bottom up. For the "Congratulations..." node, the script reads:

```
int StartingConditional()

{

return GetLocalInt(OBJECT_SELF,"NW_L_TALK-

TIMES")==0;

}
```

This checks for a local variable, named "NW_L_TALKTIMES", assigned to Aribeth, and makes sure its 0. That local variable hasn't been set yet, so it is automatically 0. Open up that node, and follow the conversation to the END DIALOGUE. Click on that, and check out the **Actions Taken** tab. You'll see a script that, among other things, assigns a value of 1 to NW_L_TALKTIMES. With this script, the NW_L_TALKTIMES switch has been flipped to on (1), ensuring that you will never see this conversation again. This script also signals the OnUserDefined event that spawns in the enemy magi.

A Word About Local Variables



We've discussed Local Variables elsewhere, but it's an important aspect of conversations, so it's worth going over again. Thing of a Local Variable as a Post-it Note that you can stick to anything in the game. You can stick them to Creatures, Objects, PC's, Areas, or even the Module itself. The text of the note can contain whatever you declared it to be. NW_L_TALKTIMES is an integer, so the Post-it Note reads a big fat 0 until a script like the one discussed above changes its value.

Whenever you want the game to remember anything, you'll need to set up a Local Variable. In this case, it remembers whether this character has been spoken to (or specifically, whether the End Dialogue of that particular conversation node has been reached). Setting up a Local Variable is as simple as opening up the Script Wizard, checking "**Set local variables**", and inputting a name for the variable and the value you want it to set to. But you'll notice the script generated by the Script Wizard differs from the Aribeth script in one important way: The Aribeth script stuck the variable to Aribeth (OBJECT_SELF). The Script Wizard script stuck the script to the player (GetPCSpeaker()). Sticking your variable on the player instead of the NPC they're speaking to is generally a good thing, since if a new player were to come and speak with this Aribeth, she would act as though she had already spoken to the new player. All she remembers is that she talked to someone; she doesn't care who. But by sticking it to the player, the corresponding script that checks the variable will scan the conversing player, and know whether or not she's spoken to that particular PC.

Of course, there are times when you want it stuck to a fixed object, like the speaker. And now is one of those times. The Academy was attacked during the conversation, and her speaking to a new player won't make the Academy be un-attacked. To give you another example, if a character were to have a nervous breakdown based on something the first NPC said, it wouldn't make since for them to be normal again when a second PC spoke to them, because the damage has already been done. That's when it's nice to stick the variable to the speaker, not the PC. Note that there is no way to stick the variable to anything other than the player with the Script Wizard. You'll have to make the change yourself with the Script Editor.

There's no limit to the amount of variables that can be assigned to an object. In fact, the character you're playing the official campaign with is covered with dozens, probably hundreds of variables. Oh, how they must itch. But remember that each variable assigned to the PC must have a different name, or it will overwrite an existing one. This isn't so much of a problem for variables assigned to NPCs, since they probably won't have more than one or two variables. In the official campaign, they use the same NW_L_TALKTIMES and NW_L_TALKLEVEL variables for virtually every character in the game.

Moving up the conversation, we have the "Listen to me..." node. This text appears only after three variables are checked: The NW_G_M1Q0CenemiesDestroyed variable, assigned to the module itself, which keeps track of how many of the spawned-in mages are dead; the NW_L_TALKLEVEL variable, another variable assigned to Aribeth; and the good old NW_L_TALKTIMES variables, also stuck to Aribeth. If the TALKLEVEL is 0, the TALKTIMES is more than 0 (which it will be after the first conversation), and 4 or more enemies are dead, this conversation will come up. As you probably guessed, the TALKLEVEL variable is assigned to 1 during this conversation, after you accept the quest. The reason they used a second variable, rather than raising TALKTIMES to 2, is probably a procedural one. This variable is doing something different (checking whether or not you've accepted a quest, rather than counting the number of times you've conversed), so it got a different variable.

Finally, the "Do not waste any more time" conversation node will only come up when 4 or more enemies are dead, the TALKTIMES is more than 0, and the TALKLEVEL is more than 0.

Tokens Versus Variables

Variables are not the only way to advance a quest. The other way is by using quest "tokens" (not to be confused with dialog tokens), inventory items that represent an accomplishment. For example, let's say the NPC Johnny Evilguy hires the players to kill another NPC, Tommy Goodboy. You accept the quest, it appears in your journal, and off you go. Obviously, you want Joe Evilguy to recognize when the players have succeeded in their mission and reward them accordingly. There are two ways to do this.

One is by having a local variable be set. Perhaps Tommy Goodboy has a custom OnDeath script that uses SetLocalInt to set the "Tommy_Is_Dead" variable, assigned to the player who killed him, to 1. When you talk to Johnny, he has a conversation node that says "I heard you killed Tommy!" that has a Starting Conditional script that uses GetLocalInt to check the status of the "Tommy_Is_dead" variable.

The other is by giving Tommy a custom item, like "Tommy's Left Testicle", marked as "Dropable." Tommy dies, the player searches his inventory, and pockets the blood-drenched body part as proof of the kill. Johnny Evilguy has a conversation node that says "That looks like Tommy's Left Testicle. He must be dead!", with a Starting Conditional that uses the **CheckPartyForItem** function to scan their inventories for the token.

You'll notice that whenever possible, the official campaign used the latter method. In fact, at any given moment in the official campaign you're probably carrying around a good half-dozen dismembered body parts. Why are the developers so fond of this method? It has many advantages:

- One less script and one less variable are required, making the debugging process easier.
- The token can be transferred to other players, and is easy for all players to keep track of.
- The token serves as a visual reminder of the accomplishment, while the variable is hidden.
- You don't need to worry about the variable being lost or written over. If the player loses the token, its his or her fault, not yours.

Also, players raised on the official campaign will be used to receiving token items during milestones in their quest, so they may expect you to use this method. But the choice is yours, and you'll want to experiment with both methods.

Taking Action in a Conversation

We've already covered one action that can take place in a conversation, the assigning of variables. But you can assign a script to a line of conversation that does just about anything. After all, that End Dialogue script from the Aribeth "Congratulations…" conversation node triggered all of the events that made the Academy get attacked, from spawning in enemies to killing instructors elsewhere in the Academy. When your mother said words could never hurt you, she clearly didn't consider the possibility that they could contain an **Actions Taken** script that spawns in a Balor.

More commonly, you'll use an Actions Taken script to turn the speaker hostile (which usually happens in response to a player saying "I'm going to kill you"), to make the speaker surrender or becoming an ally, to give the player an item or experience, or to open a store. All of these common functions can be done with the Script Wizard. To do multiple functions, just check multiple boxes. They'll all be added to the script that the wizard generates.

Give Rewards: You can choose to give any amount of gold and/or XP to the speaker. If you'd like the reward to be shared among the party, click the To Party box. To give an item, enter the item's ResRef blueprint name (under Item Properties: Advanced) and click Add. Remember that this is generating a new copy of this item in the PC's inventory, and it won't actually delete it from the speaker's inventory, if it's there at all.

Set Local Variables: As discussed above, this is where you can set a local variable so the PC or NPC will remember having had this conversation, or set or change any variable that can be retrieved by scripts assigned to any other object in your module.

Perform an Action: If you check "**No Action**," you'll be able to move the Modify Faction slider. This will change how the speaker's faction feels about the PC by the selected amount, possibly putting the speaker into hostile range. If you check "**Attack**" the faction's opinion of the PC will automatically be adjusted by minus 100, and combat will begin immediately. If you check "**Open Store**," you'll be able to enter the tag of your store into the "Script" field, and that store will automatically open at that point. To set up a Store, use the **Merchant Wizard** in the Wizards menu. Then open up its **Merchant Properties** window, and hit the Inventory button to start dragging and dropping the store's item stock.

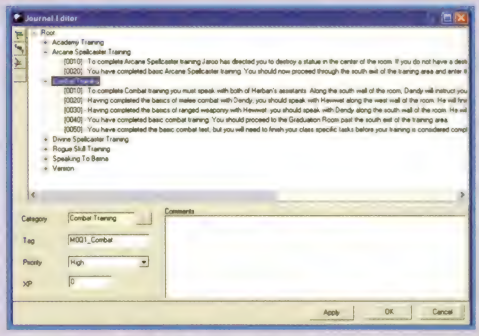
Journals & Other Actions

At any point in a conversation you can have the speaker play an animation or sound selected from pull down menus in the **Other Actions** tab. You can also have the player's journal be updated at that point– just select the Category name and the event ID number you want it to be set to. We'll cover the Journal Editor in more detail below.

Finally, in the **Current File** tab, you have two script handlers that fire at the end of the conversation. The Normal one fires when the conversation ends at an "End Dialogue", and the Aborted handler fires if the player hits Escape to bail out of the conversation midstream. Aribeth has a very meaty End Conversation Script apparently designed to ensure the attack on the Academy occurs if you bail out of the conversation after the variables have been set but before the attack happened. Aribeth is one of only a very few characters in the official campaign with an End Conversation Script. It is unlikely that you would need them in your own module.

The Journal Editor

The Journal Editor is the final tool in our toolset, and one of the simplest. As you can see, it is organized similarly to the Conversation Editor. Each node is a different quest, with each branch being a level of progression in the quest. Each branch has an ID number, which is referenced by the function that adds the entry to your journal. It's good to have your ID numbers go up by 10's instead of single digits, because it's easy to add an entry between 10 and 20, but impossible to add anything between 1 and 2. The order is important, since the journal can only be updated to an entry with a higher ID than its current state. That way if you get an update, then speak to the character who gave you the original journal entry, it won't revert to the original entry.



At the node level, you can set the **Tag** (used in script functions), the **Priority** (used to determine how high it appears in your journal screen), and the **XP** value for when its completed. At the branch level, if you check **Finish Category**, the quest will be moved to the completed folder after reaching that branch, and the player will receive the XP reward.

The player's journal can be updated in two ways. In a conversation, it's a simple matter of entering the Category and desired ID number in the Other Actions tab of the conversation branch where the quest is given. To update a journal as part of any other script, you'll need to use the **AddJournalQuestEntry** function.

Other Tips for the Conversation Editor

- Grey lines of a conversation are links that lead to a branch in a different conversation tree within the same file. To make such a link, right click on the branch that you want to copy, choose **Copy**, and then select the branch that will be the parent of the link and choose **Paste as Link**. This is tremendously useful when branches of a conversation loop back to a list of questions ("is there anything else you wanted to ask?"), or you have different branches of player dialog (perhaps based on a Status Conditional that checks the player's Intelligence or Charisma) that do not change the speaker's response.
- If you open the **Speaker Tag** pull-down menu, you'll find a list of all the objects in the area. To change the speaker of a line from the Owner (the NPC who has this designated as their dialog file) to someone else in the room, choose their tag here. When the conversation happens in the game, the chosen object's portrait will appear, giving the impression the chosen object had spoken the line.
- Do you ever just want to lay the structure of your conversation and then fill in the dialog later? To make the Input Text box stop popping up every time you Add a new branch, open the Options menu and uncheck the "Show popup when creating a new text entry" box.

- When pressed, the purple buttons at the top of the conversation editor will highlight all the conversation nodes that have a **Comment** (in the Comment tab), **Action** (in the Actions Taken tab), **Quest** (Journal entry), **Animation** (in the Other Actions tab) or **Sound** (in the Other Actions tab).
- Dialog "tokens," such as <boy/girl>, <he/she>, <class>, etc, will automatically insert the appropriate value in the game. To see a list of available tokens, right click in the Text window or Input Text pop-up and choose "**Insert Token**."

5 Thanks For Reading This Book!

We could write a chapter about every script function in *Neverwinter Nights*, but I'm sure you'd rather be making modules than reading 6,000 pages of documentation. And frankly, we'd rather be making them ourselves than writing about it. No book could cover everything this amazing piece of software is capable of, and we had no intention of trying to.

Instead, consider this to be a general overview to the art of world building. If there's anything you want to do that this book didn't teach (and there will be thousands of things, I'm sure), remember that the best teaching tool is the official campaign itself. If all you came away with after reading this book was the ability to deconstruct how the official campaign and other modules were made, then I think our job is done. With that knowledge, you should be able to find examples of nearly anything you want to do, reverse engineer them, and make them your own. Once you have that down, the only limitation is the outer limit of your own ingenuity.

If you get a chance, please check out our Haven module, soon to be available at versusbooks.com (and any other *NWN* community sites that will offer it). We'll try to fill it with helpful comments, and we'd love to hear what you think of it. You can reach us with comments, suggestions, and (hopefully) news of your own completed modules at NWN@versusbooks.com. As rabid *Neverwinter Nights* fans ourselves, we can't wait to see what you guys come up with!

Appendix "C": C Scripting and Neverwinter Nights

1 Introduction

One of the nicest features of the *Neverwinter Nights* Toolset is that the scripts that drive the module and its myriad effects and events are written in the C computer language. That's a good thing because C is one of the oldest and most popular programming languages today. However, for those of you who aren't familiar with the language, this poses a problem. Reading one of these scripts is like trying to read Sanskrit; nearly impossible. (Which is probably why you are reading this and not designing scripts!)

The purpose of this Appendix is to briefly introduce those parts of C commonly used by the scripts in *NWN*. It is not intended to teach the language but to train you to recognize and understand important aspects of a *NWN* script. After reading the C language constructs explained below and studying the examples in this appendix, you can decipher nearly any script in the game, given enough time and effort on your part.

Note

If you really want to dig into this stuff, I recommend any basic C Programming book at your local bookstore or perhaps a class in C Programming.

Like all languages, C is made up of parts. Instead of nouns, C has variables. Instead of verbs, C has functions. All we are concerned with (in terms of understanding or writing a *NWN* script) are the following parts:

- Basic script structure
- Functions
- Variables and data types
- Expressions
- Flow control

Let's hit these one by one. It cuts down on the confusion and gives you an excuse to only read what you care about. At the end, several more advanced scripts are presented, giving you the chance to practice your new skills.

2 Basic Script Structure

Every script in *NWN* has the same type of structure, composed of three main things: the return type, the name and any parameters of the script, and the body of the script (containing the script code).

Example Script

```
//this is the basic script structure

void main( )

{

….…body of the script…

…/* this is a comment inside of the script body*/

};
```

The first part of the first line (in blue) is called the **return type** (by default, this appears in a light blue). This tells you what kind of value is returned when the script finishes execution. The most common types for *NWN* scripts are:

void – The script returns nothing upon completion.

int – The script returns an integer value (like 3, 6813, or -12) upon completion.

float – The script returns a real number (a number with a fractional value, like 3.5 or -56.01) upon completion.

Note

Frequently, green text (the default color for comments in the Script editor) is found in *NWN* scripts. A comment either indicates useful information about the script or code that is ignored by the script compiler (or "turned off"). As seen above, a comment is always preceded by a // or enclosed within a /* comment body */.

The second part is the name of the script body and any parameters required by the script. This part commonly consists of two values: a **main** type returning **void**, or a **StartingConditional** type returning an **int**.

The **void main( )** script type is the most common and usually causes effects or actions to be taken when the script is executed. The following script performs a simple effect, based on a few conditions. See what you can figure out from the code below. I'll meet you on the other side with the answer:

Example Script

```
void main( )

{

    object oPC = GetEnteringObject( );

    if(GetIsPC(oPC) &&

       GetMaxHitPoints(oPC) == GetCurrentHitPoints(oPC))

    {

        ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectDamage(1),oPC);

    }

}
```

By now, you should be able to recognize the basic structure of the script (the **void main** and the curly braces). However, the code inside introduces a few concepts not discussed, yet. So don't feel stupid if you couldn't figure it out.

Here's what's going on. First, the script gets the object currently entering the area (using **GetEnteringObject( )**). The object is then checked to see if it is the PC (using **GetIsPC(oPC)**) and if the current HP of the object (still undetermined but possibly the PC) are equal to the maximum HP of the object. **If** the object is a PC object, and the maximum HP is equal to the current HP, **then** the object is given a free point of damage (using **ApplyEffectToObject( )** and **EffectDamage( )**)! Sounds like a nice little area to stay away from, doesn't it?

The **StartingConditional** type is commonly used in NPC conversations to setup conditions for the start of a conversation (as the name implies) with a PC. The return value can indicate if the NPC has spoken with the PC before or at what point in a complicated conversation they are currently at. For example, a conversation script can be written that only exposes a certain part of the NPC's conversation the first time a PC and an NPC meet.

The following script is taken from the initial conversation of an NPC you meet in the first part of the single-player campaign:

Example Script

```
int StartingConditional( )

{

    int bCondition = GetLocalInt(OBJECT_SELF, "NW_L_TALKTIMES") == 0;

    return bCondition;

}
```

This script is used by the game to determine if the NPC has spoken with the PC before. This is accomplished by checking the value of a local variable (**NW_L_TALKTIMES**). If the value is 0 (indicating they have never met), the script returns the value 1; otherwise, it returns 0 (indicating they have met before).

Declaring and Using Variables

One of the handiest features available to *NWN* scripts is the ability to declare and use local variables in the game. For example, you could declare a local variable in your module that is 0 when a certain door should be unlocked and 1 when that same door is unlocked. You could then write a script that would use the value of that variable to lock/unlock the door. If you do any scripting you will **quickly** realize how important this little feature is!

There is a family of script functions provided by BioWare that implement this ability:

GetLocalInt and **SetLocalInt** – Gets and sets a local variable of type **integer**.

GetLocalFloat and **SetLocalFloat** – Gets and sets a local variable of type **float**.

GetLocalObject and **SetLocalObject** – Gets and sets a local variable of type **object**.

GetLocalString and **SetLocalString** – Gets and sets a local string variable.

And here's a quick example of how to set and get a local integer variable stored in a module (you can tell it's in the module because of the call to **GetModule( )**):

Example Script

```
int nCurVal=0;


//the first parameter is the object storing the variable

//the second parameter is the name of the variable, and the last parameter is the value

//

SetLocalInt(GetModule(), "CF_CaveDoor_Locked", 1);

//

//the first parameter is the object storing the variable

//the second parameter is the name of the variable

//

nCurVal= GetLocalInt(GetModule(), "CF_CaveDoor_Locked");
```

First, we set the local variable **CF_CaveDoor_Locked** (located in the module object) to 1 with a call to **SetLocalInt( )**. The value is then retrieved with a call to **GetLocalInt( )**. For more examples, check out the other script examples.

Note

I know I just threw some new stuff at you but it is discussed in a later section, Expressions. For now, just soak up as much as you can and move on. You can always revisit these examples later.

The third part is the body of the script (containing the code to be executed). It always begins with a '{' and ends with a '}'. These braces (referred to as curly braces or curly brackets) bracket the script code.

The following script (taken from the official *NWN* campaign) is of the first type, main type returning void. Can you figure out what the body of the script consists of?

Example Script

```
void main( )

{

     object oPC = GetEnteringObject( );


     if (GetIsPC(oPC)&& (GetLocalInt(oPC, "NW_JOURNAL_ENTRYJOUR_M1Q0Training") < 10))

     {

          AddJournalQuestEntry("JOUR_M1Q0Training", 10, oPC, TRUE, TRUE);

          object oPC= GetNearestCreature(CREATURE_TYPE_PLAYER_CHAR,

                                         PLAYER_CHAR_IS_PC);

     }

}
```

Notice that the sample contains more than one pair of curly braces. This is a good test of recognizing which braces enclose the main script body. The main body is everything contained by the outer braces, in this case, the ones we highlighted in red.

This script (also taken from the single-player campaign) is of the second type, **StartingConditional** type returning **int**.

Note

The C language has a simple grammatical rule that is never broken and is always applied. The rule is this: every line of code must end with a semicolon. This tells the Script Editor that the line of code is complete. It works exactly like a period in a sentence; telling the reader that this sentence is finished. If we didn't have this, the code would be incomprehensible to the Script Editor.

Example Script

```
int StartingConditional( )

{

     int bCondition = GetLocalInt(OBJECT_SELF, "NW_L_TALKLEVEL") >= 1;

     return bCondition;

}
```

In this case, the value of the **bCondition** variable (discussed in a later section) is returned to the caller of the script for use in their code. In case you're curious, the script above returns 1 if the NPC and the PC have spoken at least once before and 0 if they haven't spoken before.

The main thing to take away from this section, in regard to determining the layout and contents of a script, is a method for quickly determining the main body of the script: by looking for the first and last curly brackets in the script. Everything between those two is the script body, which can be modified or used by you in another script.

3 Functions

A function is one or more lines of code performing a specific task. This task can consist of one action (one line of code) or many actions (several lines of code) that accomplish an effect.

The structure of a C function consists of a return type (for scripts, usually an integer or an object in the game), a function name, and a list of zero or more parameters.

Note

Every function call must have a pair of parentheses after the name, even if the function takes no parameters. This is a C language thing and something you will see a lot.

The following samples illustrates these parts:

Example Function

```
object GetIsPC(object oPC);
```

In the example above, the **GetIsPC( )** function returns a value of type **object** (this type can represent any object in the game), and requires one parameter (**oPC**, also of type **object**). Therefore, every time you see this function in a script, an object variable name is always present, enclosed within a set of parentheses:

Example Script

```
void main( )
{
 //Notice that the parentheses are still used in the code below, even though there are no
parameters.
 object oPC = GetEnteringObject( );

 if (GetIsPC(oPC))
 {
       //do something with the PC
 }
}
```

Functions are the heart and soul of the *NWN* scripts. They can create effects from a magical beam of light, spotlighting a holy priest, to releasing a deadly cloud of poison gas into the face of your trusted Rogue, NightShade!

Note

In fact, at this point you should realize that scripts are simply giant functions that you write. Every script you open in NWN is a giant function, usually either a main type or a StartingConditional type! You can prove this by looking back at every sample used so far. Therefore, once you master function reading and writing, you have taken a big step in using and writing scripts.

The *NWN* Toolset provides a list of functions that do just about anything you can imagine. You can pull up a full listing of the functions in Neverwinter Nights in the Script Editor's palette. [AppC01.tif]

In *NWN*, scripts commonly call several of these functions. Each function is identified by a name (such as **GetIsPC**) and *always* followed by a list of zero or more parameters, bracketed by a pair of parentheses. These parameters provide additional information used by the function when it executes. In some cases, a function has another function call "nested" inside of the pair of parentheses. This simply means that the nested function is called first, and then passes its result onto the outer function as a parameter.

For instance, what functions are called in the following script:

Example Script

```
void main( )

{

    if(GetIsPC(GetEnteringObject( )))    //check the entering object to see if it is a PC

    {

        SetPlotFlag(GetEnteringObject( ),TRUE);    //set the Plot flag of the PC object.

    }
```

The answer is three, with **GetEnteringObject( )** called twice.

First, the script returns the object entering the area, with a call to **GetEnteringObject( )**. Remember, the nested function is executed (or read) first. It then checks to see if the object is a player character (PC), using the **GetIsPC( )** function and the value returned by **GetEnteringObject( )**. If the check is successful, it makes the PC immune to harm by calling **SetPlotFlag( )**, a function that toggles on and off the "plot" aspect of objects that makes them invulnerable.

Predefined NWN Script Functions

The functions provided by the *NWN* Toolset can be divided into several groups, based on functionality. These groups are listed below, with some actual functions listed as examples:

Table 1: NWN Function Groups

Group Name	Purpose	Example function
Get functions	Retrieves objects in the current campaign, current variable values, and other values.	**GetObjectFromTag, GetMaxHitpoints, GetEnteringObject**
Set functions	Sets the value of objects in the current campaign, current variable values, and other values.	**SetTime, SetWeather, SetLocalInt**
Action functions	Perform various actions (such as using a feat or opening a door) on objects in the current campaign.	**ActionAttack, ActionCastSpellAtObject, ActionEquipMostDamagingMelee**
Effect functions	Causes various effects on objects in the current campaign.	**EffectBeam, EffectDeath, EffectTimeStop**

There is another group, Miscellaneous, containing all the functions that were not in the previous groups. You were surprised, right?

Note

To get a quick description of a function and its parameters in the Script Editor, double-click the function name. A short description appears in the lower pane.

Default Parameters

If you do any digging around in the *NWN* functions, you will eventually notice one or more parameter names followed by an '=' sign and a value. For instance:

Example Script

```
// Adds oHenchman as a henchman to oMaster

void AddHenchman(object oMaster, object oHenchman=OBJECT_SELF)
```

In the sample above, **oHenchman** demonstrates this. This means that if you do not provide a value for this parameter when calling the function (**AddHenchman(objMaster);**) the value is automatically set to **OBJECT_SELF**. This is referred to as a default parameter. In this case, because this is a function commonly called within the script of an NPC who will become the henchman, the owner of the script is usually passed as the value of the henchman object.

4 Variables and Data Types

Variables are a valuable part of the C language because they store numerical values and act as placeholders in complex expressions. In order to use a variable, it must first be declared. A variable declaration tells the C compiler the name of the variable and type of data to be stored.

Note

By default, the Script Editor displays all variable types (like int, float, and object) in blue. Use this style to quickly find the variables declared in a script.

The most common variable types in a script are:

- **integers** – identified by the **int** keyword, with some examples being 34, 6813, and -12.
- **reals** – identified by the **float** keyword, with some examples being -34.75, 10.5, and 5600.5.
- **constant strings** – identified by a quoted series of characters, with some examples being **"NW_BLDRAGON001"**, **"NW_IT_BLUEKEY"**, and **"M1Q0_Chair1"**. Unlike the other variables, string constants (named because they never change) are previously defined by *NWN*. Therefore, you just have to reference them when using them in your scripts.
- **objects in the game** – identified by the **object** keyword, with some examples being a PC entering the room, a chair in the corner of a room, or an Ancient Blue dragon, sitting in the dark and waiting for the next "adventurer".

Here are a few common examples of variable declarations:

Example Variable Declarations

```
object curObj; //shorthand for the current object being used

int nMaxHP; //shorthand for the maximum HP for a specific PC

float nDCResult; //shorthand for the result of a recent Difficulty Class roll
```

In the previous example, the variable names **nMaxHP** and **curObj** are just made-up names. When you declare your variables, you can name them whatever the hell you want. For example, if you examine the official *NWN* campaign scripts, you will notice that BioWare usually names their objects **oObject** or **pcObj** and their integers **nInteger** or **nCurrent**. This makes it easy to remember what the variable type is, later in the script. But it isn't a rule, and you can name and capitalize them however you want.

Here are a few examples of them being used in a script:

Example Script

```
//oStudent is declared and initialized in the same step. This is very common in NWN

scripts.

//nNth is an integer type variable storing our current place in the group of students.

//In addition, the string constant "M1Q0AStudent" is used here.

//

object oStudent = GetObjectByTag("M1Q0AStudent",nNth);


//bCondition is declared and initialized with the result of a logical

//comparison (more on those later) between the result of GetLocalInt and 0.

//

int bCondition = GetLocalInt(OBJECT_SELF, "NW_L_TALKTIMES") == 0;
```

As the samples demonstrated, you can easily find declarations of a variable by looking for blue type words (like **int** or **float**) immediately followed by some name. As mentioned previously, variables can be used to store the results of calls to other functions, like getting the object that just entered an area of the module:

Example Script

```
object oPC = GetEnteringObject( ); //these are commonly found in area scripts
```

Or they can be used to simplify a complex expression, making it more readable for the person examining the script:

Example Script

```
object oStudent = GetObjectByTag("M1Q0AStudent");

KillAndReplace(oStudent); //sounds vicious, doesn't it?
```

In this example, the object (representing a student) is stored in the **oStudent** variable and subsequently used in a call to **KillAndReplace( )**.

The KillAndReplace Function

KillAndReplace is a excellent function cooked up by the BioWare team. It takes an existing NPC or monster and kills them, leaving a body behind. Casey loves this function as the following disturbing e-mail snippet shows:

...Right now I'm desperate, DESPERATE for an OnSpawn script that will kill creatures and NPCs but not make the bodies disappear. As you know, my life is very corpse-oriented and I find it virtually impossible to create any game areas without a healthy dusting of corpses.

Anyway, back to the function. It is declared on the **area_m1q0b_4** script, located in the **Prelude** module of the single player campaign. It is as follows (with my comments):

Example Script

```
//The function name is KillAndReplace. It returns nothing and takes one parameter, oMark.
//
void KillAndReplace(object oMark)
{
 if(GetIsObjectValid(oMark))    //checks to make sure it exists in the game
 {
  SetPlotFlag(oMark,FALSE);    //Allows the object to be killed, even if essential to the
plot.
  AssignCommand(oMark,SetIsDestroyable(FALSE,FALSE));
  AssignCommand(OBJECT_SELF,
                                    ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectDeath(
),oMark));
 }
}
```

The first call to **AssignCommand( )** sets up the object so that the body does not fade away after death (the first "false" parameter) and that it can't be resurrected (the second "false" parameter). The second call kills the object instantaneously.

As you can see, variable declarations are easy and, if named wisely, tell you what they store. In the next section, we'll talk about expressions. Combined with variables, they are a very powerful part of scripting.

5 Expressions

In addition to function calls, you will also run across many expressions. Expressions are lines of code that perform various operations (using assignment, arithmetic, and logical operators) on one or more parts (also referred to as **arguments**) of the expression. An operator performs the actual arithmetic, assignment, or logical operation in the expression. These arguments (parts of the expression other than the operator) can be variables declared earlier, return values from a function call, or constant numerical values.

For instance, the following code adds the value of two variables (the **arguments**), and then assigns the result (a new **argument**) to a third variable (the last **argument**):
The following table lists the most common operators found in scripts. The more complex operators are discussed in greater detail afterwards.

Example Expression

```
int x;


//the '+' operator has two arguments: 7 and 10. The '=' operator has one
//argument: the sum of 7 and 10, or 17.
x=10 + 7;
```

Table 2. Common C Operators

Operator	Name	Description
=	simple assignment	Assigns a numerical value to a variable.
*, /, +, -	basic arithmetic	Performs basic arithmetic operations on two values.
++, —	increment, decrement	Increments or decrements a value by 1.
<, <=, >, >=	relational operators	Determines if the value of the first argument is less than, less than or equal, greater than, and greater than or equal, to the value of the second argument.
!	logical negation	Returns the opposite logical value of the argument.
==, !=	logical equality	Determines if two arguments are logically equal or not equal to each other. **&&** logical AND Returns **true** if both arguments are **true**; otherwise it returns **false**.
||	logical OR	Returns **true** if either argument is **true**; otherwise it returns **false**.

If you just blew a fuse on this table, and all the geek speak, don't worry. There are many examples coming up and in other parts of this appendix.

A Assignment Expressions

Commonly, a script calls several different functions. The results (or return values) indicate the current state of the game. If you didn't store that result in some way, you would have to call the function every time you needed that value. This can get tedious and is an inefficient style of programming.

A better method uses a variable to store the result. It can then be used in an expression, in place of the function call, at any later point in the script.

The following example declares a simple variable (called **oWP**, of type **object**), and then stores the result of a call to **GetObjectByTag( )**:

Example Script

```
//stores a waypoint object in the game, identified by the WP_M1Q0EgobSpawn tag

//

object oWP = GetObjectByTag("WP_M1Q0EGobSpawn",nNth);
```

Assignment expressions, like the example above, are littered throughout a script. They are essential in understanding what the script is attempting to do. In addition to assignment expressions, there are expressions for incrementing/decrementing of a variable, determining the logical relationship of two expressions, and performing basic arithmetic operations.

B Increment/Decrement Expressions

In some scripts, the value of a variable is increased (incremented) or decreased (decremented) according to some action. For instance, if you wanted to change several objects slightly (for instance, make them vulnerable in the game) in a group, you could set up a variable to keep your place as you go through every object in the group (commonly called a counter), modifying each in the same manner.

The following code demonstrates this by making each student in a group vulnerable to attacks or damage in the game.

Example Script

```
int nIdx;    //this stores our current place in the group of students


nIdx = 0;    //at the beginning, we start at 0

oGraduate = GetObjectByTag("M1Q0C_Graduate",nIdx);    //gets a specific student in a group


//We will talk about this crazy "while" thing in a later section (Flow Control)

//

while (GetIsObjectValid(oGraduate)) //this checks the object just retrieved for validity

{
```

```
    SetPlotFlag(oGraduate,FALSE);    //can now be killed, by a cruel and harsh world

    nIdx++;    //setting up to get the next student

    oGraduate = GetObjectByTag("M1Q0C_Graduate",nIdx);

//

//Notice that this line of code is exactly the same as the previous call to

GetObjectByTag( ).

//The beauty of counters!

}
```

In addition to the increment operator, there is a decrement operator. It decreases the current value of a variable by 1.

Logical Negation Expressions

In most cases, when two variables are compared to each other, the numerical value is the part being compared. If variable **x** is equal to 7 and variable **y** is equal to 10, then one can say that **x** is greater than **y**. If **y** had been equal to 7, then the two variables would be considered equal.

In addition to numerical comparison, there is another type of comparison: logical. Unlike numerical comparison, the result of a logical comparison is either **true** or **false**. Logical comparison is like asking someone a simple **yes** or **no** question. The answer is either **yes** or **no**. In other words, all or nothing.

When comparing values in this fashion, logical operators are used. If you remember, we discussed these briefly in an earlier table. One of these operators is called a logical negator. Basically, it takes the result of a logical operation (remember, the result is either 0 or 1) and turns it into the opposite. Therefore, if a **true** value (in other words, 1) is returned, the '**!**' operator would make it **false** (a value of 0).

Note

One quirk of the C language is that true and false are represented as numerical values. In other words, when code in a script logically compares two variables, the result is either 0 (false) or non-zero (true). This makes it easier to write code that could receive either numerical data or the result of a logical test. This is possible because the two types of results are both numerical. For easy reading (and writing), I will use 0 or 1 when talking about false or true. This is not exactly correct because the logical value true can also be represented by either positive or negative number. The only number that won't work is 0 (which is interpreted as false).

For example, the following code sample first calls **GetPCSpeaker( )**. This function returns the PC (of type **object**) currently in conversation with the NPC. The returned object is then used as a parameter for a call to **GetIsObjectValid( )**. This function returns **true** (or 1) if the object is a valid object; otherwise **false** (or 0). Finally, the return value is logically negated (for example, if originally **true**, it's now **false**).

Example Script

```
int StartingConditional( )

{

    return !GetIsObjectValid(GetPCSpeaker( )) ;

}
```

This is commonly used to determine if the object executing the script is currently in a conversation with a valid PC. If not, they are free to either start one with a nearby PC, or perform some action (that would seem odd if the NPC was actually in a conversation).

Logical AND/OR Expressions

These expressions are a bit trickier. They are used to determine if two expressions are logically equal.

Scripts in *NWN* use two logical operators:

- **(x && y)** - this returns **true** if the left and right expressions are *both* **true**, otherwise **false**.
- **(x || y)**- this returns **true** if *either* the left or right expression is **true**, otherwise **false**.

Often these expressions are used to determine if the rest of a script body is executed or not. For instance, the following code sample returns **true** if the object is in conversation with a valid PC **and** the object has never talked to the PC previously.

Example Script

```
int StartingConditional( )

{

    int bCondition = GetIsObjectValid(GetPCSpeaker( )) &&

            GetLocalInt(OBJECT_SELF, "NW_L_TALKLEVEL") == 0;

    return bCondition;

}
```

Let's walk through this one step-by-step to nail the concept down. First, a call is made to **GetPCSpeaker( )**. The object retrieved is then validated with a call to **GetIsObjectValid( )**, passing the returned object as a parameter. The result of this validation is the **first** argument.

The second argument is created with the following steps. First, the current value of the local variable, **NW_L_TALKLEVEL,** is retrieved with a call to **GetLocalInt( )**. This value is then logically compared to 0 (in other words, has the conversation started yet?).

The result becomes the **second** argument. These two arguments are then logically compared, using the logical AND operator. The result is **true** only if both arguments were **true**. The result is **true** only if the speaker is a PC (first argument) and the conversation was never started (the second argument). That's a lot of explaining for 3 lines of code!

Note

Because we are creating a logical expression, we only care if the conversation was started or not. Therefore, if the value is 0 the conversation hasn't started and if it is any other number we don't want to restart the conversation at the beginning.

Digging Into the GetLocalInt( ) Function

The **GetLocalInt( )** function returns the current value of the specified local variable for a certain object. The function requires two parameters: the object whose local variable we're checking and the name of the local variable being checked.

In the sample above, **NW_L_TALKLEVEL** is a local variable storing the current conversation level (the current point in an ongoing conversation) of the NPC. An NPC uses this variable to track what parts of a conversation have been "exposed" to the PC. If the value is 0, the NPC hasn't previously talked with the current PC. Values other than 0 indicate the current point in the conversation, determining the next part of the conversation that occurs.

6 Flow Control

When a script executes, the order is from the top down. In other words, the first line is executed, then the next, and the next, and so forth until it reaches the end (always indicated by a '}' bracket). This is referred to as the **flow of execution**. This is fine when you are doing something only once, or rarely, in a script. However, there are times when you'll want to do the same thing several times in a row or perform a similar action in different situations. This is where flow control comes in.

For the purposes of reading and scripting in *NWN*, there are three types of flow control:

- The **if-else** statement
- The iteration statement (using either the **while** or **for** keyword)
- The **switch** statement

There are other types of flow control in the C language but only these three are used in scripts. Each has it's own structure and behavior.

A The if - else Statement

Similar in purpose to a yes or no question, this statement tests an assumption (or the answer to the question). **If** the assumption is true, then execute the first line of code; **else** execute the second line of code. Not too hard to figure out how they got the name, is it?

The structure is as follows:

Structure of an if-else Statement

```
if(expression)
  //first line of code
else
  //second line of code
```

Note

You don't have to use both parts of the if-else statement. If you leave out the "else" portion, nothing will happen if the first assumption is incorrect.

For example, the following code checks for a valid target by calling **FindTarget( )**. It then checks to see if the target returned was valid (using **GetIsObjectValid( )**). If the target is valid, it is attacked; otherwise all actions for the object doing the search are cleared:

Example Script

```
object oTarget;

oTarget = FindTarget(fFollowDistance);
if( GetIsObjectValid(oTarget) )
{
    ClearAllActions( );
    ActionAttack(oTarget);
}
else
{
    ClearAllActions( );
}
```

In the above example, the flow of execution always executes the first two lines, and then either:

- Executes the **if** body (the clear actions and attack), skipping the **else** body.
- Skips the **if** body, executing the **else** body (clears all actions).

B The Iteration Statement

As demonstrated earlier, there are times when you want to execute the same code a certain number of times. You can do this by using a **for** or **while** loop.

The for loop executes the code predetermined number of times. Usually a variable is used to store the number of iterations. The structure is as follows:

Structure of an Iteration Statement

```
for(expression1, expression2, expression3)

{

  //code to execute

};
```

The initialization expression, **expression1**, is executed before the code in the body of the loop. Secondly, the **expression2** is evaluated and, if true, then the code in the body of the loop is executed. Finally the update expression, **expression3** is executed. The whole thing repeats itself but the second time through **expression1** is skipped. The process ends when **expression2** is no longer true.

To make sense of all this, examine the following code sample and try to figure out what is going on:

Example Script

```
int nIdx;

for(nIdx = 1;nIdx <= 5;nIdx++)

{

  SignalEvent(GetNearestObjectByTag("Ritual_Zombie" +

    IntToString(nIdx)), EventUserDefined(680));

}
```

Now that you have looked at it, let's walk through the code step-by-step.

First, a variable is created (**nIdx**), storing how many zombies have been signaled. Then the **for** loop signals the next 5 zombies (**nIdx <=5**) with a call to **SignalEvent( )**. The two parameters for the **SignalEvent( )** call are created as follows:

- The current value of the index (**nIdx**) is converted to a string using the **IntToString( )** function (for instance, the numerical value 3 becomes the string "3") and combined with the tag for a zombie object ("**Ritual_Zombie**"). The result is **"Ritual_Zombie3"**. This tag is the first parameter.
- For the second parameter, the **EventUserDefined( )** function returns the event to be run (specified by 680).

Once the parameters are created, the **SignalEvent( )** function is called, with the result that the retrieved zombie does whatever event 680 specifies. This is then repeated for four more zombies.

C The switch Statement

The final flow control statement is one of the most powerful and common statements used in *NWN*. Switch statements take as input the current value of a variable. Then, depending on the current value, code is executed. Think of it as a kind of telephone call router. Depending on the number being called (the variable value), the phone servicing that number (the code assigned to that value) is rung (executed).

The structure is as follows:

Structure of a switch Statement

```
switch(variable)
{
  case x: expression1; break;
  case y: expression2; break;
  case z: expression3; break;
  default: expression4; break;    //the default part is optional
};
```

It looks complicated (because it is) but a simple example should help clear things up. Before reading ahead, try to figure out what is going on with this code:

Example Script

```
int nSpell;

switch(Random(9))
{
    case 0: nSpell = SPELL_ENDURE_ELEMENTS;
                  break;
    case 1: nSpell = SPELL_DAZE;
                  break;
    case 2: nSpell = SPELL_MAGE_ARMOR;
                  break;
    case 3: nSpell = SPELL_MAGIC_MISSILE;
                  break;
    case 4: nSpell = SPELL_RAY_OF_ENFEEBLEMENT;
                  break;
    case 5: nSpell = SPELL_SCARE;
                  break;
    case 6: nSpell = SPELL_PROTECTION_FROM_EVIL;
                  break;
    case 7: nSpell = SPELL_NEGATIVE_ENERGY_RAY;
                  break;
    case 8: nSpell = SPELL_RAY_OF_FROST;
                  break;
}
ActionCastSpellAtObject(nSpell,GetNearestObjectByTag("M1Q0BSpellTarget"),
                                                    METAMAGIC_ANY,TRUE);
```

Were you able to figure this one out? Basically, this code randomly chooses one 1st level spell from a group of nine, and then casts it at a nearby target. You see this in the Prelude module from the single-player campaign. Let's take a closer look.

First, a random number is generated from 0 to 9, and then passed to the **switch** statement. The switch statement begins comparing the value it was passed with each case value. As soon as a match is found (for example, 5), the code for that case is executed. In this example, the value of the variable **nSpell** is set to **SPELL_SCARE** (more on this later). The **break** statement causes the switch statement to quit and the flow of execution moves to the **ActionCastSpellAtObject( )** line of code. This beast of a function actually has SEVEN parameters, but only the first two are important to us. We know this because there are defaults provided for the other five. Remember, many parameters have a default value that will automatically be entered if the script writer leaves them blank. For our purposes, the two interesting parameters are: the name of the spell and the target of the spell (retrieved by a call to **GetNearestObjectByTag( )**).

The most common usage of a **switch** statement is determining which custom action to perform in the **OnUserDefined** script for a *NWN* module object. You can find several examples of this in chapter 4.

7 Scripts Deciphered

In this section, several scripts are presented (taken from the *NWN* campaign) as an exercise to the reader. Study each script and try to figure out the action or actions caused by the script. After each script I will provide a line by line explanation, which you can use to check your answer. Remember to use all your new techniques for determining what the important bits are and when they are run by the *NWN* game engine. In addition, the rest of the chapters in this book contain many other examples of a higher level and can be used as additional quizzes. For bonus points, try to figure out which event handler the script is designed for. Good luck!

Example Script

```
M1q4mugger01_2 script

void main()

{

     object oTarget = GetLastPerceived();


     if (GetLastPerceptionSeen() == TRUE)

     {

          if (GetIsEnemy(oTarget))

          {

               ClearAllActions();

               ActionMoveAwayFromObject(oTarget,TRUE);

          }

          // If no mark has been set, save this mark and check him out.

          else if (oTarget == GetNearestCreature(CREATURE_TYPE_PLAYER_CHAR,
PLAYER_CHAR_IS_PC) &&

                    GetLocalInt(OBJECT_SELF,"NW_L_FindMark") == 0)

          {

               SetLocalObject(OBJECT_SELF,"NW_L_PCMark",oTarget);

          }

     }

}
```

I chose this one because it uses a very common technique to retrieve the current PC. But first let's see what it does.

The script is designed for the **OnPerceived** handler of an NPC mugger. This is indicated by the initial call to **GetLastPerceived( )**, retrieving the last object perceived by the mugger. This function is always used within an **OnPerception** script.

This object is then checked for enemy status with a call to **GetIsEnemy( )**. If the object is an enemy, the mugger immediately moves away, after clearing any actions the mugger might have had during the perception test.

However, if the object is not an enemy the object is checked (using **GetNearestCreature( )**) against the nearest game object with the following characteristics:

CREATURE_TYPE_PLAYER_CHAR – Looks for creatures of type player character.

PLAYER_CHAR_IS_PC – A second parameter that further narrows the creature type. In this case, the actual PC.

If that check is TRUE and the mugger has not found a mark (indicated by a local variable NW_L_FindMark) then the current PC object becomes the mugger's "mark", indicated by setting the NW_L_PCMark local variable of the mugger to the PC object.

The ultimate effect of this script is to tag the PC object as a mark for the mugger, who then follows and attempts a mugging (implemented by other scripts). The exact character implementing this script is the NPC Neibor, and his location is the Docks area in Chapter 1.

Okay, on to the next:

Example Script

```
M1q1apyre_d
void main()
{
    int nEvent = GetUserDefinedEventNumber();
    switch(nEvent)
    {
        case 115:
            object oWP = GetWaypointByTag("WP_M1Q1APyre");
            ActionMoveToObject(oWP);
            ActionPlayAnimation(ANIMATION_LOOPING_GET_LOW);
            ActionDoCommand(SignalEvent(GetArea(OBJECT_SELF),EventUserDefined(116)));
            ActionWait(10.0f);

ActionEquipItem(GetItemPossessedBy(OBJECT_SELF,"NW_IT_TORCH001"),INVENTORY_SLOT_LEFTHAND);
            ActionPlayAnimation(ANIMATION_LOOPING_GET_LOW);
            ActionDoCommand(ApplyEffectAtLocation(DURATION_TYPE_INSTANT,
                                                  EffectVisualEffect(VFX_IMP_FLAME_M),

GetLocation(GetWaypointByTag("WP_M1Q1ABody"))));
            ActionDoCommand(SignalEvent(GetArea(OBJECT_SELF),EventUserDefined(117)));
            ActionWait(10.0f);
            ActionMoveToObject(GetWaypointByTag("WP_PyreStart"));
            ActionDoCommand(DestroyObject(OBJECT_SELF));
        break;
    }

}
```

How did you do on this one? This is another nice example of a common script type (it is from the **OnUserDefined** event handler of a spawned city guard in Chapter 1, in the City Core of Neverwinter. Similar to the previous example, the script type can be guessed easily because the first call is to **GetUserDefinedEventNumber( )**, a function we learned about in the Flow Control section. After the guard is spawned into the area map (using the **OnUserDefined** script **m1q1aarea4**), the custom event 115 (shown above) is immediately fired to the guard's **OnUserDefined** script.

First, the guard walks over to his waypoint, and then an animation is played of the guard crouching low. A custom event 116 is then fired to the area's **OnUserDefined Script**, causing a plague victim to appear next to the guard and causing the guard to pause for 10 seconds.

Then a torch (retrieved from the inventory of the guard with a call to **ActionEquipItem( )**) is placed in the guard's left hand. The crouching animation is played again to get the guard back down, and then the newly spawned plague victim is lit with a call to **ActionDoCommand( )**. This function is chock full of good stuff, so let's check it out.

The first parameter is a call to **ApplyEffectAtLocation( )**. It's parameters are as follows:

DURATION_TYPE_INSTANT – The duration of the effect, in this case a flame flaring up from the effect location.

EffectVisualEffect(VFX_IMP_FLAME_M) – The type of effect, in this case a typical medium-sized flame.

GetLocation(GetWaypointByTag("WP_M1Q1ABody")) – The location where all this action is occurring, in this case the waypoint **WP_M1Q1Abody** (where the plague victim body was just spawned in).

Remember that previous lessons taught you that this nested code is executed from the inside out. Therefore, the order of execution for this **ActionDoCommand( )** call is as follows:

EffectVisualEffect( ), the second parameter

GetWaypointByTag( ), the parameter for the call to GetLocation( )

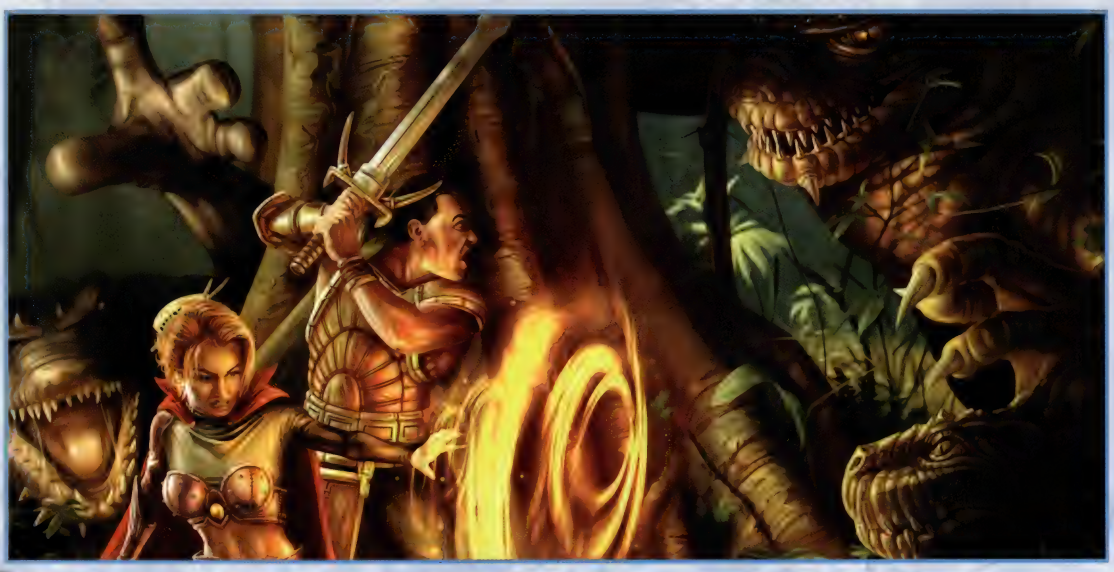
GetLocation( ), the third parameter

ApplyEffectAtLocation( ), the only parameter of the original call to ActionDoCommand( )

After the fire is lit another custom event (117) is fired to the area's **OnUserDefined** Script, causing a medium fire to appear and burn the corpse. After 7 seconds, the corpse object is destroyed ("burnt up".) and then a second after that, the medium fire is destroyed.

Finally, the guard stands there for 10 seconds and then walks back toward the gate and disappears.

Did your head explode yet? That was a pretty complicated script but provides a neat little scene to play out for the PC, increasing the dark and intense mood of the plague-ridden city. If you are interested in the custom event code for the other events, check out the **OnUserDefined** script **m1q1aarea**, located in the **City Core** area.



MONSTER APPENDIX

ANIMALS

GROUP: Bears

STRATEGY: Bears range widely in challenge rating, from the lowly Black Bear to the monstrous Ancient Dire Bear, a creature with stats that seem more in line with with Old and Ancient Dragons. Of course, bears are a heck of a lot easier to deal with than most powerful creatures, since they have no pesky spells, spell resistances, or immunities. Instead, they pretty much stand there and try to whack you with their claws. But when you add in the strength bonus and such feats as Cleave, Improved Critical and Power Attack, those claws can shred an unprotected mage in a turn or two. The simple, tank-like nature of bears makes them ideal animal companion for Druids, Wizards and Sorcerers (Summon Creature VI nets you a nice meaty Dire Bear).

SOCIETY: Always true neutral, Bears are solitary creatures (they occasionally travel in pairs) that roam the forest in search of things to put in their mouths. Black Bears just want to be left alone, but Browns and Dire Bears will pick a fight to protect their habitats. Kodiaks are found in coastal areas, and Polar Bears are found only in the far north.

ANCIENT DIRE BEAR

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 17 | HIT POINTS: 300 | ARMOR CLASS: 17

Dire Bears are larger, more vicious versions of their common cousins. They are often found in the company of powerful rangers or druids.

				SAVES	
STR	31/10	INT	3/-4	FOR	12
DEX	13/1	WIS	12/1	REF	9
CON	19/4	CHA	10/0	WIL	5

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Monster Abilities:** 1 x Intensity **Spells:** N/A

EQUIPMENT

Claw 1: Claw2d4 **Claw 2:** Claw2d4 **Special Attack:** Bite2d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

BLACK BEAR

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 19 | ARMOR CLASS: 13

A forest-dwelling omnivore, the common Black Bear is not aggressive by nature, though it can be very dangerous when protecting its cubs or food supply.

				SAVES	
STR	19/4	INT	3/-4	FOR	5
DEX	13/1	WIS	12/1	REF	4
CON	15/2	CHA	6/-2	WIL	2

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature); Weapon Focus (creature); Weapon Specialization (creature) **Combat Feats:** Cleave **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

BROWN BEAR

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 5 | HIT POINTS: 51 | ARMOR CLASS: 15

More aggressive and carnivorous than the smaller Black Bear, Brown Bears are commonly considered to be very bad-tempered and territorial.

				SAVES	
STR	27/8	INT	3/-4	FOR	9
DEX	13/1	WIS	12/1	REF	6
CON	19/4	CHA	6/-2	WIL	3

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature); Weapon Focus (creature); Weapon Specialization (creature) **Combat Feats:** Cleave **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Dire Bear

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 9 | HIT POINTS: 102 | ARMOR CLASS: 17

Dire Bears are larger, more vicious versions of their common cousins. They are often found in the company of powerful rangers or druids.

STR	31/10	INT	3/-4	FOR	12
DEX	13/1	WIS	12/1	REF	9
CON	19/4	CHA	10/0	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature); Skill Focus (Discipline); Skill Focus (Listen); Skill Focus (Spot) **Combat Feats:** Cleave **Monster Abilities:** 1 x Intensity, Improved **Spells:** N/A

EQUIPMENT

Claw 1: Claw2d4 **Claw 2:** Claw2d4 **Special Attack:** Bite2d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Kodiak Bear

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 6 | HIT POINTS: 76 | ARMOR CLASS: 15

STR	27/8	INT	3/-4	FOR	10
DEX	13/1	WIS	12/1	REF	7
CON	19/4	CHA	6/-2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Polar Bear

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 6 | HIT POINTS: 68 | ARMOR CLASS: 15

Their thick, white fur a product of their northern climate, these graceful and powerful hunters are a rarity, even in their natural habitat.

STR	27/8	INT	3/-4	FOR	10
DEX	13/1	WIS	12/1	REF	7
CON	19/4	CHA	6/-2	WIL	3

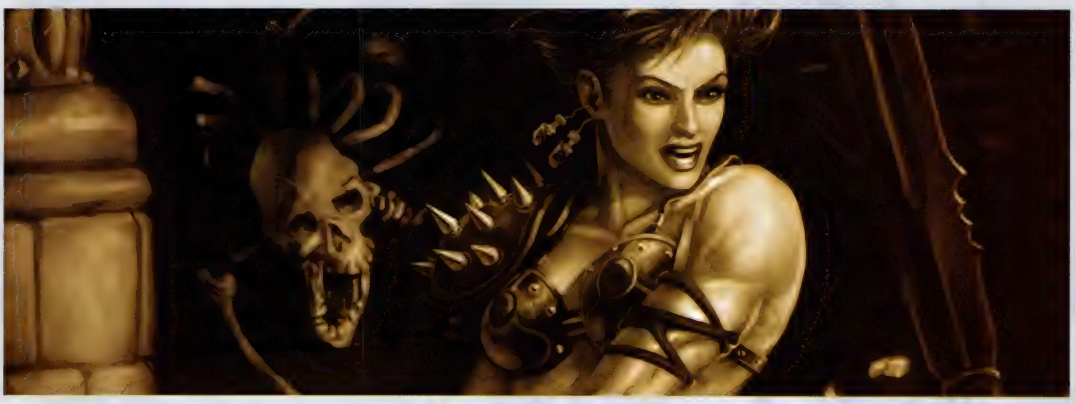
SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d8 **Skin/Hide:** N/A **Standard Equipment:** N/A



GROUP: Birds

STRATEGY: In addition to the three listed below, you can make a Penguin by taking a Chicken and changing its Appearance (in the Basic tab) to "Penguin." Now your Polar Bears have something to eat! Birds are basically just scenery, but a good murder of hostile ravens (remember to change the faction–they default to "commoner") could be a fun encounter for Level 1 players.

Chicken

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/4 | HIT POINTS: 4 | ARMOR CLASS: 10

This flightless, domesticated bird is prized both for its tender meat and for the eggs it lays on an almost daily basis.

				SAVES	
STR	3/-4	INT	3/-4	FOR	1
DEX	7/-2	WIS	8/-1	REF	0
CON	8/-1	CHA	6/-2	WIL	0

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Standard Equipment:** N/A

Falcon

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/3 | HIT POINTS: 5 | ARMOR CLASS: 10

Falcons are majestic birds, often trained by adventurers to be loyal companions. Their superior eyesight and speed are their greatest assets.

				SAVES	
STR	6/-2	INT	3/-4	FOR	2
DEX	9/-1	WIS	14/2	REF	1
CON	10/0	CHA	6/-2	WIL	2

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Standard Equipment:** N/A

Raven

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/3 | HIT POINTS: 5 | ARMOR CLASS: 10

With their cold eyes and glossy black feathers, these airborne scavengers are often considered ill omens or harbingers of death.

				SAVES	
STR	3/-4	INT	3/-4	FOR	2
DEX	7/-2	WIS	14/2	REF	0
CON	10/0	CHA	6/-2	WIL	2

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Standard Equipment:** N/A

GROUP: Canines

STRATEGY: Individual wolves, even the tank-like Pack Leader, are fairly simple to dispatch. They have few special properties and only one attack, but they tend to attack in large packs. Area effect spells are very useful, but spell-casters will want someone to hide behind or their relentless attacks will fizzle the spell for sure. Winter Wolves can cast Cone of Cold three times a day, so stand way back and don't forget the Flame Arrows.

SOCIETY: **Dogs:** Make sure to give your domestic dogs conversation files so Druids and Rangers will have a reason to feel special. **Wolves:** Wolves and Dire Wolves should always attack in large packs (5 - 10), so they're more dangerous than their Challenge Rating suggests. **Winter Wolves:** Though they're listed as Chaotic Neutral in *Neverwinter Nights*, these cunning, intelligent predators are traditionally Neutral Evil. They're smart enough to speak common and often team up with other evil creatures of the icy northern reaches, such as Frost Giants. **Worgs:** Moderately intelligent and generally evil, Worgs are often found in the service of Goblins.

Dire Wolf

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 5 | HIT POINTS: 51 | ARMOR CLASS: 14

Dire Wolves are larger, more vicious versions of their common cousins. They are often found in the company of powerful Rangers or Druids.

STR	25/7	INT	3/-4	FOR	8
DEX	15/2	WIS	12/1	REF	7
CON	17/3	CHA	10/0	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature); Weapon Focus (creature) **Combat Feats:** Cleave **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Dog

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1 | HIT POINTS: 15 | ARMOR CLASS: 10

Easily domesticated and trained, the common Dog has been a popular guard, hunting servant, and traveling companion for ages.

STR	15/2	INT	3/-4	FOR	5
DEX	11/0	WIS	12/1	REF	3
CON	15/2	CHA	6/-2	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Pack Leader

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 13 | HIT POINTS: 250 | ARMOR CLASS: 14

Dire Wolves are larger, more vicious versions of their common cousins. They are often found in the company of powerful Rangers or Druids.

STR	25/7	INT	3/-4	FOR	8
DEX	15/2	WIS	12/1	REF	7
CON	17/3	CHA	10/0	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature); Weapon Focus (creature) **Combat Feats:** Cleave **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Winter Wolf

RACE: M. Beast | ALIGNMENT: CN

CHALLENGE RATING: 5 | HIT POINTS: 51 | ARMOR CLASS: 15

Dangerous predators of the northern reaches, Winter Wolves pursue their prey relentlessly. Their breath is said to be cold enough to shatter stones.

				SAVES	
STR	18/4	INT	9/-1	FOR	8
DEX	13/1	WIS	13/1	REF	6
CON	16/3	CHA	10/0	WIL	3

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 3 x Cone of Frost **Spells:** N/A

EQUIPMENT

Claw 1: Bite1d8 **Claw 2:** Bite1d8 **Special Attack:** Bite1d8 **Skin/Hide:** Frost Giant/Winter Wolf Prop. (Damage Vulnerability: Fire 50% Dmg. Vuln.; Immunity: Dmg. Type: Cold 100% Imm. Bon.) **Standard Equipment:** N/A

Wolf

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1 | HIT POINTS: 15 | ARMOR CLASS: 10

An icon of rugged individualism, yet renowned for its ability to hunt in packs, the noble Wolf is an animal of both mystique and power.

				SAVES	
STR	13/1	INT	3/-4	FOR	5
DEX	11/0	WIS	12/1	REF	3
CON	15/2	CHA	6/-2	WIL	1

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

Worg

RACE: M.Beast | ALIGNMENT: CN

CHALLENGE RATING: 3 | HIT POINTS: 30 | ARMOR CLASS: 14

Worgs are powerful wolves with an evil disposition and an uncanny intelligence. They are sometimes found with Goblins, whom they serve as mounts and guardians.

				SAVES	
STR	17/3	INT	6/-2	FOR	6
DEX	15/2	WIS	14/2	REF	6
CON	15/2	CHA	10/0	WIL	3

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Bite1d6 **Claw 2:** Bite1d6 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

STRATEGY: High Dexterity scores and the Weapon Finesse feat ensure that while cats don't generally do a lot of damage, they usually hit. Fortunately, they're usually encountered alone, and are often indifferent to players. Crag Cats can be dangerous for low-level players, since they can't be charmed and their 4d4 bite attack can be a game ender. Dire Tigers don't need Weapon Finesse because their strength is obscene; their quick, deadly strikes make Summon Creature V one of the best 5th level spells for Clerics, Sorcerers, Wizards and Bards.

SOCIETY: Cats are usually solitary hunters, so it would be unusual to see more than one in an area map. Cougars and Crag Cats prowl mountainous regions, jaguars and panthers prefer dense jungle, and lions and leopards prefer the savannah. Dire Tigers are rare, but can be found just about anywhere, even in the underdark.

Cougar

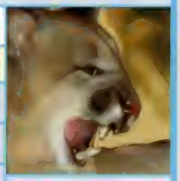
RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 22 | ARMOR CLASS: 15

A large and powerful wild cat, the Cougar makes its home in mountainous regions. It's often identified by its tawny, unmarked coat.

STR	16/3	INT	3/-4	FOR	5
DEX	19/4	WIS	12/1	REF	7
CON	15/2	CHA	6/-2	WIL	2

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

Crag Cat

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 22 | ARMOR CLASS: 16

The fabled "Hunter-of-Men" is a sure-footed predator of the North. More than just a simple bobcat, however, it wields some magical properties as well.

STR	16/3	INT	13/1	FOR	5
DEX	19/4	WIS	12/1	REF	7
CON	15/2	CHA	6/-2	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite4d4 **Skin/Hide:** Crag Cat Properties; Immunity: Spell School: Enchantment **Standard Equipment:** N/A

Dire Tiger

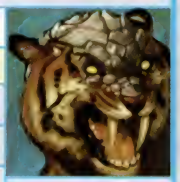
RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 11 | HIT POINTS: 120 | ARMOR CLASS: 18

Dire Tigers are larger, more vicious versions of their common cousins. They are often found in the company of powerful Rangers or Druids.

STR	27/8	INT	3/-4	FOR	13
DEX	15/2	WIS	12/1	REF	12
CON	17/3	CHA	10/0	WIL	6

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw2d4 **Claw 2:** Claw2d4 **Special Attack:** Bite2d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

Jaguar

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 17

With similar markings to the Leopard, the Jaguar is a common predator in densely forested regions where its natural camouflage allows for the greatest stealth.

STR	16/3	INT	3/-4	FOR	6
DEX	19/4	WIS	12/1	REF	8
CON	15/2	CHA	6/-2	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Leopard

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 19 | ARMOR CLASS: 15

STR	16/3	INT	3/-4	FOR	5
DEX	19/4	WIS	12/1	REF	7
CON	15/2	CHA	6/-2	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

Lion

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 32 | ARMOR CLASS: 16

Known as the King of the Great Cats, the Lion is large, fierce, and regal. A confident hunter, it attacks swiftly and with crushing force.

STR	21/5	INT	3/-4	FOR	6
DEX	17/3	WIS	12/1	REF	7
CON	15/2	CHA	6/-2	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Panther

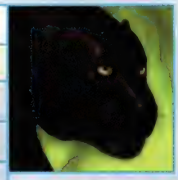
RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 19 | ARMOR CLASS: 15

Black as a Chultan night and just as deadly, these sleek cats attack from the shadows, disappearing back into them once the hunt is through.

STR	16/3	INT	3/-4	FOR	5
DEX	19/4	WIS	12/1	REF	7
CON	15/2	CHA	6/-2	WIL	2

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** Claw1d2 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

STRATEGY: Most of the Animals in this category are beasts of burden or prey for larger Animals. Only Boars and Badgers are likely to be of interest to players, since weak but agile Dire Badgers and powerful Dire Boars can be made Animal Companions by the spells Summon Creature I and II, respectively.

GROUP: Other

Badger

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/2 | HIT POINTS: 6 | ARMOR CLASS: 10

The common Badger is a small but fierce mammal. Its strong forelimbs are armed with long claws, used for both digging and self-defense.

				SAVES		
STR	8/-1	INT	3/-4		FOR	4
DEX	7/-2	WIS	12/1		REF	0
CON	15/2	CHA	6/-2		WIL	1

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** Claw1d2 **Special Attack:** Bite1d3 **Skin/Hide:** N/A **Standard Equipment:** N/A

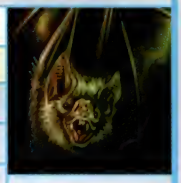
Bat

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/3 | HIT POINTS: 5 | ARMOR CLASS: 11

These winged, nocturnal rodents are rarely considered threatening in their own right, though traditional folklore does associate them with vampirism and the undead.

				SAVES		
STR	3/-4	INT	3/-4		FOR	2
DEX	8/-1	WIS	14/2		REF	1
CON	10/0	CHA	4/-3		WIL	2



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Standard Equipment:** N/A

Boar

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 23 | ARMOR CLASS: 16

Though not true carnivores, these wild swine are very bad-tempered and usually charge anyone who disturbs them, ripping at them with their tusks.

				SAVES		
STR	15/2	INT	3/-4		FOR	6
DEX	10/0	WIS	13/1		REF	3
CON	17/3	CHA	4/-3		WIL	2

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Gore1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Cow

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/6 | HIT POINTS: 5 | ARMOR CLASS: 7

Raised for both its hearty meat and nourishing milk, these large herbivores are a staple resident of rural barnyards.

				SAVES		
STR	10/0	INT	3/-4		FOR	2
DEX	5/-3	WIS	10/0		REF	0
CON	10/0	CHA	6/-2		WIL	0

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Standard Equipment:** N/A

Deer

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/4 | HIT POINTS: 5 | ARMOR CLASS: 10

Typically docile in nature, these wild, skittish herbivores gather in small groupings that scatter quickly when threatened.

STR	10/0	INT	3/-4	FOR	2
DEX	11/0	WIS	12/1	REF	2
CON	10/0	CHA	10/0	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Gore1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

Dire Badger

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 25 | ARMOR CLASS: 17

Dire Badgers are larger, more vicious versions of their common cousins. They are often found in the company of powerful Rangers or Druids.

STR	10/0	INT	3/-4	FOR	4
DEX	17/3	WIS	12/1	REF	6
CON	12/1	CHA	4/-3	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

Dire Boar

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 5 | HIT POINTS: 52 | ARMOR CLASS: 16

Dire Boars are larger, more vicious versions of their common cousins. They are often found in the company of powerful Rangers or Druids.

STR	27/8	INT	3/-4	FOR	8
DEX	10/0	WIS	13/1	REF	5
CON	17/3	CHA	8/-1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** 1 x Ferocity **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d8 **Skin/Hide:** N/A **Standard Equipment:** N/A

Ox

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/2 | HIT POINTS: 6 | ARMOR CLASS: 9

These heavy draft animals are prized possessions that not all farmers can afford. Oxen are well-suited to pulling plows and milling grain.

STR	12/1	INT	3/-4	FOR	3
DEX	5/-3	WIS	10/0	REF	0
CON	12/1	CHA	6/-2	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Gore1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

White Stag

RACE: Animal | ALIGNMENT: N

CHALLENGE RATING: 1/4 | HIT POINTS: 5 | ARMOR CLASS: 10

Typically docile in nature, these wild, skittish herbivores gather in small groups that scatter quickly when threatened.

STR	10/0	INT	3/-4	FOR	2
DEX	11/0	WIS	12/1	REF	2
CON	10/0	CHA	10/0	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Gore1d6 **Skin/Hide:** N/A **Standard Equipment:** N/A

CONSTRUCTS

GROUP: Constructs

STRATEGY: Constructs are among the most annoying enemies in the game, since their tough hides negate many forms of attack. As a general rule, you'll need to stick with bludgeoning weapons and elemental spells (fire, cold, etc.) to harm them.

All **Golems** have some form of damage reduction, so make sure your weapon is at least a +1 for Clay and Flesh Golems, +2 for Bone and Steel, and +3 for Iron Golems. Clay and Flesh Golems are immune to piercing and slashing weapons, so bust out the hammers, maces, and flails. Remember that slings are bludgeoning weapons (which any class can use), and you should have encountered a billion or so enchanted sling bullets by the time you run into these nasty foes. All Golems are slow (although Clay Golems have Haste) and have few ranged attacks (Iron Golems have Poison Gas), so you can keep your distance and fire off a hail of arrows, bolts, bullets and spells. Or just run past them and hope their attack of opportunity misses.

Shield Guardians have many of the defensive abilities of Golems, plus Regeneration, plus complete immunity to spells Level 4 and lower, plus a handful of offensive spells (Fireball, Magic Missile, and Melf's Acid Arrow).

The **Horrors** are less tough on defense, but more capable on offense. Both Battle Horrors and Helmed Horrors wield a Great Sword +1 that does 3 extra fire damage in their capable hands. Battle Horrors have 4 more Strength points and the Cleave, Weapon Specialization and Power Attack feats, but are otherwise the same as Helmed Horrors.

Minogons have attributes similar to the horrors, but are slightly less powerful on offense. They can do a stunning Howl attack three times a day.

SOCIETY: Golems: Golems are created by powerful wizards and master craftsmen in a process that takes months, a small fortune, and a chunk of the creator's XP. Therefore, they are available only to the most powerful and wealthy wizards. All Golems are mindless automatons that unquestioningly follow the instructions of their creators. They're usually used for menial labor and personal defense, since they aren't bright enough to do much else.

Shield Guardians: Like Golems, Shield Guardians are expensive to create and available only to elite wizards. They are controlled by an amulet worn by the wizard, and will cease to function if the amulet is destroyed. They can cast the spells that were cast into them by their creator, so feel free to change the spell list for any Shield Guardians you create for your campaign.

Battle Horror/Helmed Horror: Any wizard can create a Golem, but only evil wizards of extreme power (level 18+) can make a Helmed Horror. These animated suits of armor are smarter than Golems and can be given more complicated tasks, such as hunting down and killing a specific individual. The secrets of their creation are believed to be lost, but many Horrors from long ago still exist, striving tirelessly to fulfill the commands of their long-dead masters.

Battle Horror

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 11 | HIT POINTS: 71 | ARMOR CLASS: 11

Battle Horrors are magically animated suits of empty armor. They have a greater array of magic powers than do their lesser kin, Helmed Horrors.

STR	20/5	INT	10/0	FOR	4
DEX	13/1	WIS	16/3	REF	5
CON	10/0	CHA	12/1	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** N/A **Combat Feats:** Cleave; Weapon Proficiency; Weapon Specialization (greatsword) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Helmed/Battle Horror Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Immunity: Specific Spell: Bad Strref, Confusion, Flame Arrow, Mage Armor; True Seeing) **Std. Eqp.:** Helmed/Battle Horror Sword: Full Plate

Bone Golem

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 11 | HIT POINTS: 77 | ARMOR CLASS: 19

Created by demented wizards, Bone Golems are generally used to guard prized possessions. Their bodies are pieced together from various creatures, both humanoid and monster.

STR	21/5	INT	3/-4	FOR	4
DEX	9/-1	WIS	11/0	REF	3
CON	10/0	CHA	3/-4	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 3 x Howl, Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam3d8 **Skin/Hide:** Bone Golem Prop. (DR: +2 Soak 20 Dmg.; Immunity: Dmg. Type: Piercing 50% Imm. Bonus, Slashing 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Standard Equipment:** N/A

Clay Golem

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 12 | HIT POINTS: 60 | ARMOR CLASS: 22

Unable to speak or make any vocal noise, these kilned and crafted horrors walk and move with a slow, clumsy gait.

STR	25/7	INT	3/-4	FOR	3
DEX	9/-1	WIS	11/0	REF	2
CON	10/0	CHA	3/-4	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** 1x Haste (Caster Level: 1)

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Clay Golem Prop. (DR: +2 Soak 20 Dmg.; Immunity: Dmg. Type: Piercing 100% Imm. Bonus, Slashing 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Standard Equipment:** N/A

Flesh Golem

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 9 | HIT POINTS: 49 | ARMOR CLASS: 18

While not true undead, these ghoulish constructs are crafted from collections of stolen human body parts, stitched together into single composite form.

STR	21/5	INT	3/-4	FOR	3
DEX	9/-1	WIS	11/0	REF	2
CON	10/0	CHA	3/-4	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 2 x Rage, Greater **Spells:** N/A

EQUIPMENT

Claw 1: Slam2d8 **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Flesh Golem Prop. (DR: +2 Soak 20 Dmg.; Immunity: Dmg. Type: Piercing 50% Imm. Bonus, Slashing 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison, Spells Level 9 or Lower) **Standard Equipment:** N/A

Helmed Horror

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 9 | HIT POINTS: 71 | ARMOR CLASS: 11

Helmed Horrors are suits of empty armor imbued with magical life and purpose. They make tireless and extremely effective guardians, soldiers, and bounty hunters.

STR	16/3	INT	10/0	FOR	4
DEX	13/1	WIS	16/3	REF	5
CON	10/0	CHA	12/1	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Helmed/Battle Horror Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Immunity: Specific Spell: Bad Strref, Confusion, Flame Arrow, Mage Armor; True Seeing) **Std. Eqp.:** Helmed/Battle Horror Sword; Full Plate

Iron Golem

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 14 | HIT POINTS: 99 | ARMOR CLASS: 30

Forged from solid iron, the crafting of these mighty juggernauts often requires the expertise and master craftwork of either dwarven blacksmiths or gnome artificers.

STR	33/11	INT	3/-4	FOR	6
DEX	9/-1	WIS	11/0	REF	5
CON	10/0	CHA	3/-4	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 2 x Iron Golem Poison Gas **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Iron Golem Prop. (DR: +3 Soak 30 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Poison) **Std. Eqp.:** N/A

Minogon

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 7 | HIT POINTS: 68 | ARMOR CLASS: 18

Forged from wizard's steel and imbued with raging spirits, Minogons are dangerous foes indeed. Their savage cries have left many of their victims stunned.

STR	21/5	INT	3/-4	FOR	5
DEX	10/0	WIS	12/1	REF	2
CON	17/3	CHA	9/-1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A**Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency **Other Feats:** Darkvision **Monster Abilities:** 3 x Howling,Stunning **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:**Gore1d8 **Skin/Hide:** Construct Prop. (Immunity: Misc.: Critical Hits,Death Magic, Disease ,Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** Minotaur/Minogon/Frost Giant Greataxe

Shield Guardian

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 12 | HIT POINTS: 82 | ARMOR CLASS: 24

Created by spellcasters and controlled by means of a special amulet, Shield Guardians are predominantly defensive in combat, protecting their masters with unswerving loyalty.

STR	22/6	INT	10/0	FOR	5
DEX	10/0	WIS	10/0	REF	5
CON	10/0	CHA	3/-4	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature)**Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** _____

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam1d8 **Skin/Hide:** Shield Guardian Prop. (Immunity: Misc.: Critical Hits, Death magic, Diease, Level/Ability Drain, Mind-Affecting Spells, Poison;Spells by Level 4 or lower;Regeneration +5) **Std. Eqp.:**N/A

Stone Golem

RACE: Construct | ALIGNMENT: N

CHALLENGE RATING: 13 | HIT POINTS: 77 | ARMOR CLASS: 26

Carved and etched with arcane symbols, Stone Golems are formidable opponents. Not only are they physically powerful, they are also very difficult to harm.

STR	29/9	INT	3/-4	FOR	4
DEX	9/-1	WIS	11/0	REF	3
CON	10/0	CHA	3/-4	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency(creature) **Combat Feats:** N/A) **Other Feats:** Darkvision **Monster Abilities:** 3 x Bolt, Slow **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Stone Golem Prop. (DR: +2 Soak 30 Damage; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

DRAGONS

STRATEGY: With such a huge assortment of cool variations to choose from, Dragons are likely to be the most common high-level encounter in the game. Going solo against a Dragon is suicide, so unless you're Level 20 and your target is merely an adult, you'll want to bring along a few friends. Before you head into the dragon's den, make sure you have a good variety of useful classes (**cough cough** not rogues **cough**) and you've discussed your strategy and prepared appropriate spell lists.

The most annoying aspect of dragons is the Spell Resistance (SR) of their hides, which acts as an Armor Class against magic. To refresh your memory on how SR works, when you attempt to cast a spell at a Dragon, a d20 is rolled and your caster level is added to the roll. If you don't beat their SR, the target is completely unaffected. So an offensive spell cast by 10th Level Wizard against a Dragon with an SR of 20 will be completely shrugged off by the Dragon 50% of the time. If it does make it past the Spell Resistance, the Dragon then gets its saving throw (which is generally very high) as normal. So offensive spells are generally not effective against Dragons.

Of course, their spell resistance won't effect spells that don't target them, so Mages, Clerics, Druids and Bards will want to empty their spell slots by casting summon spells like Summon Creature, Planar Binding, and Create Greater Undead; gift the Fighters with defensive spells like Ethereal Visage, Protection From Elements, Shadow Shield, Energy Buffer and Spell Resistance; and beef up the whole party with mass enhancement spells like Magic Circle Against Alignment, Aura Against Alignment, War Cry, and Mass Haste. In the actual battle, high-level Druids can kick ass with Nature's Balance, Premonition, Elemental Swarm and Shapechange. Clerics should cast Divine Power on themselves and pretend they're Fighters. Arcane Spellcasters can do the same thing with the little-used Tenser's Transformation, or, if they can cast 9th Level spells, tend to the party during Time Stops. Bards can sing. Rogues can stand far away and reorganize their inventory.

After the flurry of enhancement-oriented spell-casting, it's up to the Fighters, Rangers, Paladins, Barbarians and Monks to do the job. Make sure to do the research on the kind of Dragon you're about to fight, so everyone can have armaments with the appropriate elemental resistance. Red, Gold, White and Brass Dragons have vulnerabilities to a specific elemental type, so an appropriately enchanted weapon would be a boon, but won't have a huge impact on the fight.

SOCIETY: Dragons in *Neverwinter Nights* come in ten flavors and three different ages. An Adult Dragon is 100-200 years old, an Old Dragon has seen 400-600 years of life, and an Ancient Dragon can have up to 1,000 years under his belt. Dragons do not stop growing at maturity like most races: They keep getting bigger, smarter, and more powerful.

Dragons differ drastically by type, but they all have a few things in common: An obsession with wealth, a certain paranoia about defending it, and the need to eat. While Dragons don't have to eat as much as their size suggests, and are able to consume some inorganic materials, their presence usually has a significant impact on the food chain of the areas where they live. For this reason, Dragons are very territorial and always live alone, unless very young. Dragons rarely cooperate with each other and Chromatic Dragons will actively seek out and eliminate competitors in their territory. All Dragons can speak Common and may enjoy conversing with Humans, to outwit them in complicated mind games (Chromatic Dragons), learn about treasure (any Dragon), or provide help (Metallic Dragons). Some can polymorph into Human or Elf form and occasionally involve themselves in the affairs of the lesser races; this has been the story basis for a great many modules.

GROUP
Chromatic Dragons
(Evil)

STRATEGY: **Black Dragons:** Black Dragons live in swamps and underground. The are always Chaotic Evil and tend to be unusually obsessed with wealth, even by Dragon standards. Their breath is acidic.
Blue Dragons: Blue Dragons are Lawful Evil. They hunt in the blue skies over deserts, or burrow in the sand to ambush prey. They live in vast underground caverns in dry, hot areas, and horde gems, particularly sapphires. Blue Dragons breath lightning.
Green Dragons: Green Dragons live in old-growth forests, and usually make their lairs in hillside caves. Green Dragons are savage killers that enjoy toying with their prey (particularly Elves), but may charm and interrogate humanoids to learn about the outside world. Their breath is acidic.
Red Dragons: Reds are the mightiest of the Chromatic Dragons, and also the greediest, most vain, and most likely to prey on humanoids (they particularly enjoy eating Human and Elven children). They live in mountain caves and have many perches where they can survey the surrounding areas. They breathe fire.
White Dragons: White Dragons are the dumbest and weakest of the Chromatic Dragons, but still powerful predators. They freeze their prey with their icy breath before consuming it, and usually live in ice caves in the far north. Frost Giants are natural (and powerful) enemies of White Dragons.

Adult Black Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 13 | HIT POINTS: 199 | ARMOR CLASS: 26

Black Dragons are evil-tempered, cunning, and malevolent. Especially fond of coins, they often question their humanoid prey about stockpiles of gold before killing them.

STR	23/6	INT	12/1	FOR	15
DEX	10/0	WIS	13/1	REF	11
CON	19/4	CHA	12/1	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid **Spells:** Caster Level 3

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d6 **Skin/Hide:** Ad. Black Dragon Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Acid 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 20 SR) **Std. Eqp.:** N/A

Old Black Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 17 | HIT POINTS: 287 | ARMOR CLASS: 32

Black Dragons are evil-tempered, cunning, and malevolent. Especially fond of coins, they often question their humanoid prey about stockpiles of gold before killing them.

STR	29/9	INT	14/2	FOR	19
DEX	10/0	WIS	15/2	REF	14
CON	21/5	CHA	14/2	WIL	16

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Old Black Dragon Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Acid 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 22 SR) **Std. Eqp.:** N/A

Ancient Black Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 21 | HIT POINTS: 387 | ARMOR CLASS: 38

Black Dragons are evil-tempered, cunning, and malevolent. Especially fond of coins, they often question their humanoid prey about stockpiles of gold before killing them.

STR	33/11	INT	16/3	FOR	23
DEX	10/0	WIS	17/3	REF	17
CON	23/6	CHA	16/3	WIL	20

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Ancient Black Dragon Prop. (DR: +2 Soak 15 Dmg.; Immunity: Dmg. Type: Acid 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Adult Blue Dragon

RACE: Dragon | ALIGNMENT: LE

CHALLENGE RATING: 14 | HIT POINTS: 241 | ARMOR CLASS: 28

Blue Dragons are vain and territorial. They prefer to hunt on cloudless days, when their blue scales allow them to blend with the open skies.

STR	27/8	INT	16/3	FOR	17
DEX	10/0	WIS	17/3	REF	12
CON	21/5	CHA	16/3	WIL	15

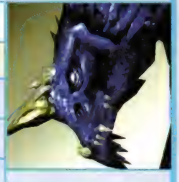
SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Lightning; 3 x Dragon Wing Buffet **Spells:** Caster Level 5

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Adult Blue Dragon Prop. (DR:+1 Soak 5 DMG.; ID: Electrical 100% IB; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 20 SR) **Std. Eqp.:** N/A



Old Blue Dragon

RACE: Dragon | ALIGNMENT: LE

CHALLENGE RATING: 18 | HIT POINTS: 337 | ARMOR CLASS: 34

Blue Dragons are vain and territorial. They prefer to hunt on cloudless days, when their blue scales allow them to blend with the open skies.

STR	31/10	INT	18/4	FOR	21
DEX	10/0	WIS	19/4	REF	15
CON	23/6	CHA	18/4	WIL	19

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency(creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Lightning **Spells:** Cast Level 9

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Old Blue Dragon Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 22 SR) **Std. Eqp.:** N/A

Ancient Blue Dragon

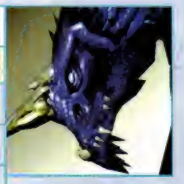
RACE: Dragon | ALIGNMENT: LE

CHALLENGE RATING: 23 | HIT POINTS: 387 | ARMOR CLASS: 38

Blue Dragons are vain and territorial. They prefer to hunt on cloudless days, when their blue scales allow them to blend with the open skies.

STR	35/12	INT	20/5	FOR	25
DEX	10/0	WIS	21/5	REF	18
CON	25/7	CHA	20/5	WIL	23

SAVES



FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Knockdown; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave; Magical Feats; Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Lightning **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite4d6 **Skin/Hide:** Ancient Blue Dragon Prop. (DR: +2 Soak 15 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 26 SR) **Std. Eqp.:** N/A

Adult Green Dragon

RACE: Dragon | ALIGNMENT: LE

CHALLENGE RATING: 14 | HIT POINTS: 230 | ARMOR CLASS: 27

Green Dragons are belligerent and prone to attacking without provocation. They make their lairs in old growth forests where they fly beneath the canopy.

STR	27/8	INT	16/3	FOR	17
DEX	10/0	WIS	17/3	REF	12
CON	21/5	CHA	16/3	WIL	15

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature); Skill Focus (search, spot) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid **Spells:** Caster Level 5

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Adult Copper/Green Dragon Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Acid100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 20 SR) **Std. Eqp.:** N/A

Old Green Dragon

RACE: Dragon | ALIGNMENT: LE

CHALLENGE RATING: 18 | HIT POINTS: 325 | ARMOR CLASS: 33

Green Dragons are belligerent and prone to attacking without provocation. They make their lairs in old growth forests where they fly beneath the canopy.

STR	31/10	INT	18/4	FOR	21
DEX	10/0	WIS	19/4	REF	15
CON	23/6	CHA	18/4	WIL	19

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Knockdown; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature); Skill Focus (listen, search, spot) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid **Spells:** Caster Level 5

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Old Copper/Green Dragon Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Acid 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Ancient Green Dragon

RACE: Dragon | ALIGNMENT: LE

CHALLENGE RATING: 23 | HIT POINTS: 432 | ARMOR CLASS: 37

Green Dragons are belligerent and prone to attacking without provocation. They make their lairs in old growth forests where they fly beneath the canopy.

STR	35/12	INT	20/5	FOR	25
DEX	10/0	WIS	21/5	REF	18
CON	25/7	CHA	20/5	WIL	23

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm, Improved Knockdown, Improved Power Attack, Knockdown; Power Attack **Class/ Racial Feats:** Alertness; Proficiency (creature); Improved Critical (creature); Proficiency (creature); Skill Focus (listen, search, spot) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities: Spells:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid **Spells:** Caster Level 13

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** Bite4d6 **Skin/Hide:** Ancient Green Dragon Prop. (DR: +2 Soak 15 Dmg.; Immunity: Dmg. Type: Acid100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 26 SR) **Std. Eqp.:** N/A

Adult Red Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 15 | HIT POINTS: 253 | ARMOR CLASS: 29

Red Dragons are the most covetous of all Dragons. Exceptionally vain, they are forever seeking ways of making other creatures pay them tribute.

STR	33/11	INT	16/3	FOR	18
DEX	10/0	WIS	19/4	REF	13
CON	21/5	CHA	16/3	WIL	17

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature); Skill Focus (listen) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Fire **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Adult Red Dragon Prop. (DR: +1 Soak 5 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 20 SR) **Std. Eqp.:** N/A

Old Red Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 19 | HIT POINTS: 378 | ARMOR CLASS: 33

Red Dragons are the most covetous of all Dragons. Exceptionally vain, they are forever seeking ways of making other creatures pay them tribute.

STR	35/12	INT	20/5	FOR	23
DEX	10/0	WIS	21/5	REF	16
CON	25/7	CHA	20/5	WIL	21

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature); Skill Focus (listen) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:Spells:** 3 x Dragon Breath, Fire **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** 4d6**Skin/Hide:** Old Red Dragon Prop. (DR: +1 Soak 10 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Ancient Red Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 23 | HIT POINTS: 527 | ARMOR CLASS: 39

Red Dragons are the most covetous of all Dragons. Exceptionally vain, they are forever seeking ways of making other creatures pay them tribute.

STR	39/14	INT	24/7	FOR	28
DEX	10/0	WIS	25/7	REF	19
CON	29/9	CHA	24/7	WIL	26

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Disarm; Improved Knockdown; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature) Proficiency (creature); Skill Focus (listen, search, spot) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:Spells:1** x Aura, Dragon Fear;3 x Dragon Breath, Fire **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** 4d6**Skin/Hide:** Ancient Red Dragon Prop. (DR: +2 Soak 15 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 28 SR) **Std. Eqp.:** N/A

Adult White Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 12 | HIT POINTS: 189 | ARMOR CLASS: 25

Among the smallest and least intelligent of Dragonkind, most White Dragons are simply animalistic predators with a predilection for gems and frozen meat.

STR	23/6	INT	10/0	SAVES FOR	15
DEX	10/0	WIS	11/0	SAVES REF	11
CON	19/4	CHA	10/0	SAVES WIL	11

FEATS & ABILITIES

Active Combat Feats: Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Cold **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite 2d6 **Skin/Hide:** Adult White Dragon Prop. (DR: +1 Soak 5 Dmg.; DV: Fire 50% Dmg. Vuln.; Immunity: Dmg. Type: Cold100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 18 SR) **Std. Eqp.:** N/A

Old White Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 15 | HIT POINTS: 276 | ARMOR CLASS: 31

Among the smallest and least intelligent of Dragonkind, most White Dragons are simply animalistic predators with a predilection for gems and frozen meat.

STR	29/9	INT	12/1	SAVES FOR	19
DEX	10/0	WIS	13/1	SAVES REF	14
CON	21/5	CHA	12/1	SAVES WIL	15

FEATS & ABILITIES

Active Combat Feats: Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature); Skill Focus (listen) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Cold; 3 x Dragon Wing Buffet **Spells:** N/A

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite 2d8 **Skin/Hide:** Old White Dragon Prop. (DR: +1 Soak 10 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Ancient White Dragon

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 19 | HIT POINTS: 375 | ARMOR CLASS: 37

Among the smallest and least intelligent of Dragonkind, most White Dragons are simply animalistic predators with a predilection for gems and frozen meat.

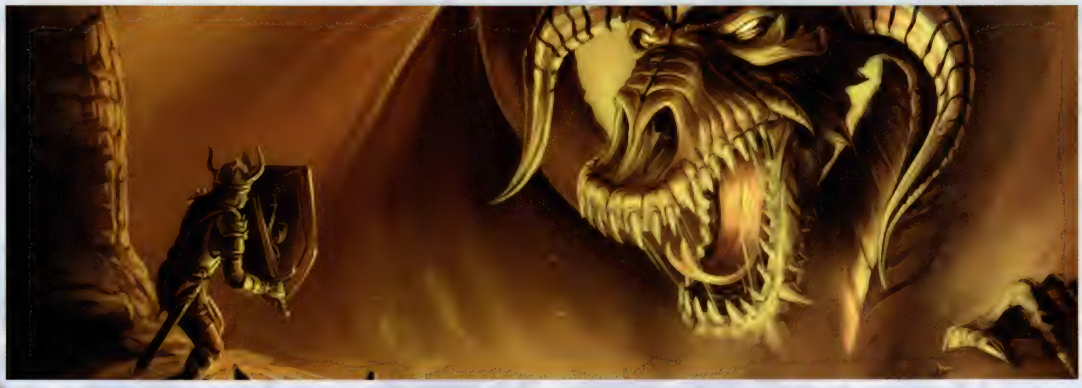
STR	33/11	INT	14/2	SAVES FOR	23
DEX	10/0	WIS	15/2	SAVES REF	17
CON	23/6	CHA	14/2	SAVES WIL	19

FEATS & ABILITIES

Active Combat Feats: Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Cold **Spells:** N/A

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite 2d8 **Skin/Hide:** Ancient White Dragon Prop. (DR: +2 Soak 15 Dmg.; DV: Fire 50% Dmg. Vuln.; Immunity: Dmg. Type: Cold100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A



GROUP
Metallic Dragons (Good)

STRATEGY: **Brass Dragons:** Brass Dragons are unusual in many ways. They spend most of their time sunbathing and rarely eat anything except dew. They share their preferred desert environment with powerful Blue Dragons, their natural enemies. Brass Dragons love intellectual stimulation and the greatest danger they pose to most adventurers is boring them to death with small talk. Brasses have two breath weapons: Fire, and sleep.

Bronze Dragons: Curious Bronzes often pose as humans and take an interest in humanoid affairs. They are friendly and believe strongly in justice, but do share the greed and vanity of their species, and will likely ask for compensation in exchange for aid. They live in coastal climes and can breathe underwater indefinitely, but make their lairs above water, usually in caves that can only be accessed from the sea. They can breath both lightning and repulsion gas.

Copper Dragons: Copper Dragons love pranks and jokes and may become frustrated with those who don't. But adventurers have little to fear if they have a good sense of humor and don't threaten a cCpper Dragon's treasure. Coppers are powerful hunters with both acid and gas breath. They live in hills and mountains, like Red Dragons, with whom they often come into conflict.

Gold Dragons: The mightiest of the metallic Dragons, Gold Dragons are wise, just, and magically lucky. They may take Human form and will involve themselves in humanoid affairs to promote goodness and justice, even without reward. They can make their home anywhere, and breathe both fire and weakening gas.

Silver Dragons: Silver Dragons are the Dragons most likely to take humanoid form, and are always cheerful and good-natured. They may live for decades in the guise of a Human, even forming friendships and subsisting exclusively on Human food. They live in mountainous regions, and often come in conflict with Red Dragons. They can breathe frost and paralyzing gas.

Adult Brass Dragon

RACE: Dragon | ALIGNMENT: CG

CHALLENGE RATING: 13 | HIT POINTS: 199 | ARMOR CLASS: 26

Brass Dragons are talkative gossipmongers. They may have useful information, but usually share it only after long rambling and hinting for a gift.

STR	23/6	INT	14/2	FOR	15
DEX	10/0	WIS	15/2	REF	11
CON	19/4	CHA	14/2	WIL	13

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Fire, Sleep **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d6 **Skin/Hide:** Adult Brass Dragon Prop. (DR: +1 Soak 5 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 26 SR) **Std. Eqp.:** N/A

Old Brass Dragon

RACE: Dragon | ALIGNMENT: CG

CHALLENGE RATING: 17 | HIT POINTS: 287 | ARMOR CLASS: 32

Brass Dragons are talkative gossipmongers. They may have useful information, but usually share it only after long rambling and hinting for a gift.

STR	29/9	INT	16/3	FOR	19
DEX	10/0	WIS	17/3	REF	14
CON	21/5	CHA	16/3	WIL	17

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:Spells:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Fire, Sleep **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite 2d8 **Skin/Hide:** Old Brass Dragon Prop. (DR: +1 Soak 10 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Ancient Brass Dragon

RACE: Dragon | ALIGNMENT: CG

CHALLENGE RATING: 22 | HIT POINTS: 418 | ARMOR CLASS: 38

Brass Dragons are talkative gossipmongers. They may have useful information, but usually share it only after long rambling and hinting for a gift.

STR	33/11	INT	18/4	FOR	24
DEX	10/0	WIS	19/4	REF	17
CON	25/7	CHA	18/4	WIL	21

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Knockdown; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear;3 x Dragon Breath, Fire, Sleep **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Ancient Brass Dragon Prop. (DR: +2 Soak 15 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 26 SR) **Std. Eqp.:** N/A

Adult Bronze Dragon

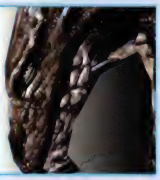
RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 15 | HIT POINTS: 241 | ARMOR CLASS: 28

Bronze Dragons are inquisitive and enjoy polymorphing into animals to better observe adventurers. Fascinated by warfare, they'll eagerly join a just cause... for good pay.

STR	27/8	INT	20/5	FOR	17
DEX	10/0	WIS	21/5	REF	12
CON	21/5	CHA	20/5	WIL	17

SAVES



FEATS & ABILITIES

Active Combat Feats: Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Gas, Lightning **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Adult Bronze Dragon Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 22 SR) **Std. Eqp.:** N/A

Old Bronze Dragon

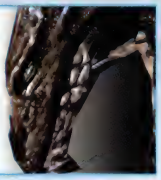
RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 19 | HIT POINTS: 337 | ARMOR CLASS: 34

Bronze Dragons are inquisitive and enjoy polymorphing into animals to better observe adventurers. Fascinated by warfare, they'll eagerly join a just cause... for good pay.

STR	31/10	INT	22/6	FOR	21
DEX	10/0	WIS	23/6	REF	15
CON	23/6	CHA	22/6	WIL	21

SAVES



FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Gas, Lightning **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Old Bronze Dragon Prop. (DR: +1 Soak10 Dmg.; Immunity: Dmg. Type: Electrical100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Ancient Bronze Dragon

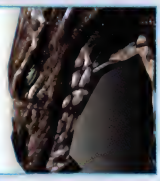
RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 24 | HIT POINTS: 445 | ARMOR CLASS: 38

Bronze Dragons are inquisitive and enjoy polymorphing into animals to better observe adventurers. Fascinated by warfare, they'll eagerly join a just cause... for good pay.

STR	35/12	INT	24/7	FOR	25
DEX	10/0	WIS	25/7	REF	18
CON	25/7	CHA	24/7	WIL	25

SAVES



FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Knockdown; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical; Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Gas, Lightning **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** Bite4d6 **Skin/Hide:** Ancient Bronze Dragon Prop. (DR: +2 Soak15 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 28 SR) **Std. Eqp.:** N/A

Adult Copper Dragon

RACE: Dragon | ALIGNMENT: CG

CHALLENGE RATING: 14 | HIT POINTS: 210 | ARMOR CLASS: 28

Copper Dragons are incorrigible pranksters, joke tellers, and riddlers. Most are good-natured but also have a covetous, miserly streak.

STR	23/6	INT	16/3	FOR	16
DEX	10/0	WIS	17/3	REF	12
CON	19/4	CHA	16/3	WIL	15

SAVES

FEATS & ABILITIES

Active Combat Feats: Knockdown; Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting; Spell Penetration **Other Feats:** Darkvision; Turn Undead **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid, Gas **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d6 **Skin/Hide:** Adult Copper/Green Dragon Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Acid100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 20 SR) **Std. Eqp.:** N/A

Old Copper Dragon

RACE: Dragon | ALIGNMENT: CG

CHALLENGE RATING: 18 | HIT POINTS: 299 | ARMOR CLASS: 33

Copper Dragons are incorrigible pranksters, joke tellers, and riddlers. Most are good-natured but also have a covetous, miserly streak.

STR	29/9	INT	18/4	FOR	20
DEX	10/0	WIS	19/4	REF	15
CON	21/5	CHA	19/4	WIL	19

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature); Skill Focus (Spot) **Combat Feats:** Cleave **Other Feats:** Darkvision,Turn Undead **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid, Gas **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Old Copper/Green Dragon Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Acid 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 24 SR) **Std. Eqp.:** N/A

Ancient Copper Dragon

RACE: Dragon | ALIGNMENT: CG

CHALLENGE RATING: 23 | HIT POINTS: 400 | ARMOR CLASS: 39

Copper Dragons are incorrigible pranksters, joke tellers, and riddlers. Most are good-natured but also have a covetous, miserly streak.

STR	33/11	INT	20/5	FOR	24
DEX	10/0	WIS	21/5	REF	18
CON	23/6	CHA	20/5	WIL	23

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature); Skill Focus (search, spot) **Combat Feats:** Cleave **Other Feats:** Darkvision, Turn Undead **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Acid, Gas **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Ancient Copper Dragon Prop. (DR: +2 Soak 15 Dmg.; Immunity: Dmg. Type: Acid 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 28 SR) **Std. Eqp.:** N/A

Adult Gold Dragon

RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 16 | HIT POINTS: 264 | ARMOR CLASS: 30

Gold Dragons are graceful, sinuous, and wise. They hate injustice and foul play, often embarking on self-appointed quests to promote good.

STR	33/11	INT	20/5	FOR	18
DEX	10/0	WIS	21/5	REF	13
CON	21/5	CHA	20/5	WIL	18

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Fire, Weaken **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d6 **Skin/Hide:** Adult Gold Dragon Prop. (DR: +1 Soak 5 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 22 SR) **Std. Eqp.:** N/A

Old Gold Dragon

RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 21 | HIT POINTS: 391 | ARMOR CLASS: 36

Gold Dragons are graceful, sinuous, and wise. They hate injustice and foul play, often embarking on self-appointed quests to promote good.

STR	39/14	INT	24/7	FOR	23
DEX	10/0	WIS	25/7	REF	16
CON	25/7	CHA	24/7	WIL	23

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Fire, Weaken **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** Bite4d6 **Skin/Hide:** Old Gold Dragon Prop. (DR: +1 Soak 10 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 26 SR) **Std. Eqp.:** N/A

Ancient Gold Dragon

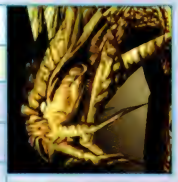
RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 26 | HIT POINTS: 542 | ARMOR CLASS: 42

Gold Dragons are graceful, sinuous, and wise. They hate injustice and foul play, often embarking on self-appointed quests to promote good.

STR	43/16	INT	28/9	FOR	28
DEX	10/0	WIS	29/9	REF	19
CON	29/9	CHA	28/9	WIL	28

SAVES



FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Knockdown; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Fire, Weaken **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** Bite4d6 **Skin/Hide:** Ancient Gold Dragon Prop. (DR: +2 Soak 15 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 30 SR) **Std. Eqp.:** N/A

Adult Silver Dragon

RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 16 | HIT POINTS: 253 | ARMOR CLASS: 29

Silver Dragons are regal and statuesque. They cheerfully assist good creatures in genuine need, often taking Human or Elven form to maintain their anonymity.

STR	27/8	INT	20/5	FOR	18
DEX	10/0	WIS	21/5	REF	13
CON	21/5	CHA	20/5	WIL	18

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Magical Feats:** Combat Casting **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Cold, Paralysis **Spells:** Caster Level 7

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Adult Silver Dragon Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Acid, Cold 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 22 SR) **Std. Eqp.:** N/A

Old Silver Dragon

RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 20 | HIT POINTS: 350 | ARMOR CLASS: 35

Silver Dragons are regal and statuesque. They cheerfully assist good creatures in genuine need, often taking Human or Elven form to maintain their anonymity.

STR	31/10	INT	22/6	FOR	22
DEX	10/0	WIS	23/6	REF	16
CON	23/6	CHA	22/6	WIL	22

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Cold, Paralysis **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Bite2d8 **Skin/Hide:** Old Silver Dragon Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Acid, Cold 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 26 SR) **Std. Eqp.:** N/A

Ancient Silver Dragon

RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 25 | HIT POINTS: 459 | ARMOR CLASS: 41

Silver Dragons are regal and statuesque. They cheerfully assist good creatures in genuine need, often taking Human or Elven form to maintain their anonymity.

STR	35/12	INT	26/8	SAVES FOR	26
DEX	10/0	WIS	27/8	REF	19
CON	25/7	CHA	26/8	WIL	27

FEATS & ABILITIES

Active Combat Feats: Disarm; Improved Power Attack; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura, Dragon Fear; 3 x Dragon Breath, Cold, Paralysis **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d8 **Claw 2:** Claw2d8 **Special Attack:** Bite4d6 **Skin/Hide:** Ancient Silver Dragon Prop. (DR: +2 Soak 15 Dmg.; Immunity: Dmg. Type: Acid, Cold 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Sneak Attack; Spell Resistance Bonus of 28 SR) **Std. Eqp.:** N/A

STRATEGY: A polymorphed Dragon can interbreed with other species, and while rare, there are those who have some Dragon blood in their veins. The two shown here were characters in the official campaign, so if your module calls for a Half- or Quarter-Dragon, you may want to make one of your own. To do so, give it a few feats and skills of its parent dragon, the appropriate breath ability, the Half-Dragon/Half-Fiend Bite and Claw as Natural Equipment, and a custom-made hide with Ability Bonuses and the appropriate immunities. Voila, a Half-Dragon. A DM who wanted to make a part-Dragon race available to his players could put all of the attributes into an unstealable, indestructible amulet (or something) and give it to his players at the beginning of the campaign.

GROUP: Half-Dragons

Half-Dragon Cleric

RACE: Dragon | ALIGNMENT: LG

CHALLENGE RATING: 12 | HIT POINTS: 84 | ARMOR CLASS: 15

STR	21/5	INT	12/1	SAVES FOR	8
DEX	12/1	WIS	17/3	REF	4
CON	12/1	CHA	14/2	WIL	10

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (morningstar) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency **Magical Feats:** Combat Casting; Spell Penetration **Other Feats:** Darkvision; Hardness vs. Enchantments; Immunity to Sleep, Keen Sense; Weapon Proficiency (Elf); Skill Affinity (Listen, Search, Spot); Turn Undead **Monster Abilities:** 1 x Cone of Frost **Spells:** Cleric Level 5

EQUIPMENT

Claw 1: Half-Dragon/Half-Fiend Bite (1d6 Dmg.) **Claw 2:** Half-Dragon/Half-Fiend Claw (1d4 Dmg.) **Special Attack:** N/A **Skin/Hide:** Silver Half-Dragon Prop. (Ability Bonus: Charisma +4, Constitution +4, Dexterity +2, Intelligence +2, Strength +4, Wisdom +4; Immunity: Dmg. Type: Cold 100% Imm. Bonus; Immunity: Misc.: Paralysis, Mind-Affecting Spells) **Std. Eqp.:** Morningstar; Noble's Tunic

Half-Dragon Sorcerer

RACE: Dragon | ALIGNMENT: CE

CHALLENGE RATING: 14 | HIT POINTS: 104 | ARMOR CLASS: 18

STR	16/3	INT	12/1	SAVES FOR	5
DEX	14/2	WIS	12/1	REF	5
CON	15/2	CHA	19/4	WIL	8

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Finesse **Defensive Feats:** Shield Proficiency; Toughness **Magical Feats:** Combat Casting; Spell Focus (Evocation) **Other Feats:** Darkvision; Quick To Master **Monster Abilities:** 1 x Cone of Fire **Spells:** Sorcerer Level 5

EQUIPMENT

Claw 1: Half-Dragon/Half-Fiend Claw (1d4) **Claw 2:** Half-Dragon/Half-Fiend Claw (1d4) **Special Attack:** Half-Dragon/Half-Fiend Bite (1d6) **Skin/Hide:** Red Half-Dragon Prop. (Ability Bonus: Charisma +4, Constitution +4, Dexterity +2, Intelligence +2, Strength +4, Wisdom +4; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Paralysis, Mind-Affecting Spells) **Std. Eqp.** Dagger; Commoner's Outfit

ELEMENTALS

GROUP
Elementals

STRATEGY: Elementals don't appear naturally in our plane, so the best way to deal with them is to kill the creature who summoned them. Base Elementals are fairly easy to defeat (although they can pack an offensive punch), but fighting a Huge, Greater or Elder Elemental forces you to deal with such difficult defensive properties as Damage Reduction up to 15/+3. Fortunately, Elementals have no spell resistance, so using an offensive spell of an element they're not immune to can be very effective. Earth Elementals are particularly vulnerable to most offensive spells, since they have terrible Reflex saves.

Air Elementals: The incredible dexterity of air elementals, combined with Weapon Finesse, makes their attacks hard to dodge and gives them a high AC. Spells are effective, but stay away from ones with a Reflex save.

Earth Elementals: Beefy Earth Elementals turn their strength bonus into incredibly damaging attacks. Stay out of their way and take them down with ranged weapons and spells that take advantage of their low reflex saves.

Fire Elementals: The only elemental to have a weakness, you'll get a 50% bonus for your cold spells and attacks.

Water Elementals: Like Fire Elementals, Water Elementals are well-rounded fighters with complete immunity to fire. They're not quite as dexterous, but are stronger and tougher.

Invisible Stalker: Invisible Stalkers are basically invisible Air Elementals. Their stats are in line with a high-level Human Fighter/Rogue, but they do everything while under the influence of an Improved Invisibility spell.

SOCIETY: With the exception of Invisible Stalkers, Elementals have very little intelligence and basically fight with abandon until the spell that summons them ends and they can return to the natural plane of existence. Invisible Stalkers are intelligent beings used not for simple combat like the others, but for more complex tasks like assassinations. Once summoned, they stay until their task is complete.

Air Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 18

Among the swiftest and most agile creatures in existence, Air Elementals are often summoned for use in great battles over vast stretches of open terrain.

STR	12/1	INT	4/-3	FOR	3
DEX	21/5	WIS	11/0	REF	6
CON	14/2	CHA	11/0	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Air Elemental Whirlwind **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam1d6 **Skin/Hide:** Elemental Prop. (Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Huge Air Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 13 | HIT POINTS: 136 | ARMOR CLASS: 23

Among the swiftest and most agile creatures in existence, Air Elementals are often summoned for use in great battles over vast stretches of open terrain.

STR	18/4	INT	6/-2	FOR	9
DEX	29/9	WIS	11/0	REF	14
CON	18/4	CHA	11/0	WIL	10

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Air Elemental Whirlwind **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d8 **Skin/Hide:** Huge/Greater Elemental Prop. (DR: +2 Soak 10 Dmg.: Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Elder Air Elemental

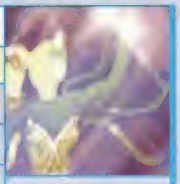
RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 19 | HIT POINTS: 204 | ARMOR CLASS: 27

Even swifter and more agile than their lesser kin, these Air Elementals are among the oldest of their kind.

STR	22/6	INT	6/-2	FOR	12
DEX	33/11	WIS	11/0	REF	19
CON	18/4	CHA	11/0	WIL	14

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Air Elemental Whirlwind **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d8 **Skin/Hide:** Elder Elemental Prop. (DR: +3 Soak 15 Dmg.; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Greater Air Elemental

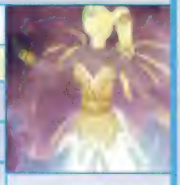
RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 17 | HIT POINTS: 178 | ARMOR CLASS: 28

Among the swiftest and most agile creatures in existence, Air Elementals are often summoned for use in great battles over vast stretches of open terrain.

STR	20/5	INT	6/-2	FOR	11
DEX	31/10	WIS	11/0	REF	17
CON	18/4	CHA	11/0	WIL	12

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Air Elemental Whirlwind **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d8 **Skin/Hide:** Huge/Greater Elemental Prop. (DR: +2 Soak 10 Dmg.; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Earth Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 30 | ARMOR CLASS: 17

Earth Elementals are immensely strong and tough, capable of pounding almost anything into rubble. They are often summoned by armies laying siege to great cities.

STR	21/5	INT	4/-3	FOR	4
DEX	8/-1	WIS	11/0	REF	0
CON	17/3	CHA	11/0	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam1d8 **Skin/Hide:** Elemental Prop. (Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Huge Earth Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 11 | HIT POINTS: 152 | ARMOR CLASS: 19

Earth Elementals are immensely strong and tough, capable of pounding almost anything into rubble. they are often summoned by armies laying siege to great cities.

STR	29/9	INT	6/-2	FOR	10
DEX	8/-1	WIS	11/0	REF	4
CON	21/5	CHA	11/0	WIL	10

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Huge/Greater Elemental Prop. (DR: +2 Soak 10 Dmg.; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Elder Earth Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 16 | HIT POINTS: 228 | ARMOR CLASS: 23

Even stronger then their lesser kin, these Earth Elementals are among the oldest of their kind.

STR	33/11	INT	6/-2	FOR	13
DEX	8/-1	WIS	11/0	REF	7
CON	21/5	CHA	11/0	WIL	14

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Elder Elemental Prop. (DR: +3 Soak 15 Dmg.: Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Greater Earth Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 14 | HIT POINTS: 199 | ARMOR CLASS: 21

Earth Elementals are immensely strong and tough, capable of pounding almost anything into rubble. They are often summoned by armies laying siege to great cities.

STR	31/10	INT	6/-2	FOR	12
DEX	8/-1	WIS	11/0	REF	6
CON	21	CHA	11/0	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Huge/Greater Elemental Prop. (DR: +2 Soak 10 Dmg.: Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Fire Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 16

Fire Elementals are fast and agile. The merest touch from their fiery bodies is sufficient to set many materials aflame.

STR	12/1	INT	4/-3	FOR	3
DEX	17/3	WIS	11/0	REF	4
CON	14/2	CHA	11/0	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Fire Elemental Slam **Skin/Hide:** Fire Elemental Prop. (Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Huge Fire Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 12 | HIT POINTS: 136 | ARMOR CLASS: 21

Fire Elementals are fast and agile. The merest touch from their fiery bodies is sufficient to set many materials aflame.

STR	18/4	INT	6/-2	FOR	9
DEX	25/7	WIS	11/0	REF	12
CON	18/4	CHA	11/0	WIL	10

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Elder/Greater/Huge Fire Elemental Slam **Skin/Hide:** Huge/Greater Fire Elemental Prop. (DR.: +2 Soak 10 Dmg.: DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Elder Fire Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 18 | HIT POINTS: 204 | ARMOR CLASS: 27

Even faster and more agile than their lesser kin, these Fire Elementals are among the oldest of their kind.

STR	22/6	INT	6/-2	FOR	12
DEX	29/9	WIS	11/0	REF	17
CON	18/4	CHA	11/0	WIL	14

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Elder/Greater/Huge Fire Elemental Slam **Skin/Hide:** Elder Fire Elemental Prop. (DR.: +3 Soak 15 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Greater Fire Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 15 | HIT POINTS: 178 | ARMOR CLASS: 26

Fire Elementals are fast and agile. The merest touch from their fiery bodies is sufficient to set many materials aflame.

STR	20/5	INT	6/-2	FOR	11
DEX	27/8	WIS	11/0	REF	15
CON	18/4	CHA	11/0	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Mobility **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Elder/Greater/Huge Fire Elemental Slam **Skin/Hide:** Huge/Greater Fire Elemental Prop. (DR.: +2 Soak 10 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Water Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 30 | ARMOR CLASS: 19

Ferocious and powerful as a stormy sea, Water Elementals are incapable of venturing far from the body of water that gave them birth.

STR	16/3	INT	4/-3	FOR	4
DEX	12/1	WIS	11/0	REF	2
CON	17/3	CHA	11/0	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Water Elemental Drown **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam1d8 **Skin/Hide:** Elemental Prop. (Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Huge Water Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 11 | HIT POINTS: 152 | ARMOR CLASS: 23

Ferocious and powerful as a stormy sea. Water Elementals are incapable of venturing far from the body of water that gave them birth.

STR	24/7	INT	6/-2	FOR	10
DEX	18/4	WIS	11/0	REF	9
CON	21/5	CHA	11/0	WIL	10

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Water Elemental Drown **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Huge/Greater Water Elemental Prop. (DR: +2 Soak 10 Dmg.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Elder Water Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 16 | HIT POINTS: 228 | ARMOR CLASS: 23

Even faster and more agile than their lesser kin, these Water Elementals are among the oldest of their kind.

STR	28/9	INT	6/-2	FOR	13
DEX	22/6	WIS	11/0	REF	14
CON	21/5	CHA	11/0	WIL	14

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse, Water Elemental Drown **Spells:** N/A

EQUIPMENT

Claw 1: Slam2d10 **Claw 2:** Slam2d10 **Special Attack:** Slam2d10 **Skin/Hide:** Elder Water Elemental Prop. (DR: +3 Soak 15 Dmg.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Great Water Elemental

RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 14 | HIT POINTS: 199 | ARMOR CLASS: 24

Ferocious and powerful as a stormy sea, Water Elementals are incapable of venturing far from the body of water that gave them birth.

STR	26/8	INT	6/-2	FOR	12
DEX	20/5	WIS	11/0	REF	12
CON	21/5	CHA	11/0	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Pulse; Water Elemental Drown **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d10 **Skin/Hide:** Huge/Greater Water Elemental Prop. (DR: +2 Soak 10 Dmg.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Invisible Stalker

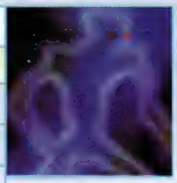
RACE: Elemental | ALIGNMENT: N

CHALLENGE RATING: 6 | HIT POINTS: 52 | ARMOR CLASS: 17

Creatures native to the Plane of Air, Invisible Stalkers are often summoned to the Prime Material to act as spies, assassins, and bounty hunters.

STR	18/4	INT	14/2	FOR	4
DEX	19/4	WIS	15/2	REF	6
CON	14/2	CHA	11/0	WIL	8

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature); Weapon Focus (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** _____

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam2d6 **Skin/Hide:** Invisible Stalker Prop. (Immunity: Misc.: Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

GIANTS

GROUP: Common & Noble

STRATEGY: Giants combine herculean strength with impressive Fortitude. They dish out a serious beating, but lack Dexterity and have low Reflex saves so they're vulnerable both to physical attacks and damaging spells. Archers and spellcasters will be in good shape, but things can turn south quickly for characters stuck in the tank role, so a defensive spell that gives damage reduction or regeneration would be a thoughtful gift from a spellcaster.

Ettins may have a slightly lower CR, but they can be very dangerous. They expertly weild two weapons at once, doubling their amount of attacks. Fire Giants are immune to Fire and weak to Cold; Frost Giants are the opposite. Otherwise, there isn't too much art to fighting Giants.

SOCIETY: Giants usually travel in bands of 6-9 members. They may also be accompanied by non-Giant species like Dire Wolves and Ogres (Hill Giants), Winter Wolves and young White Dragons (Frost Giants), Hell Hounds and Trolls (Fire Giants), Brown Bears, Orcs, and Goblins (Ettins).

Ettins tend to live in dark underground caves and remote craggy areas. They usually travel alone, or with goblins and orcs. Occasionally they form small gangs ruled by an unusually strong leader.

Hill Giants and Mountain Giants live in… Um… Hills and mountains. I kid you not. They are aggressive to outsiders but occasionally trade with outsiders.

Fire Giants dwell in areas of extreme warmth, where volcanoes and hot springs abound. Frost Giants live in the frigid northerlands. Both live in large groups ruled by a monarch and use long-term military strategies to subjugate other species in the area where they live. In addition to companion races, they may have slaves of other races fighting by their sides.

Ettin

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 8 | HIT POINTS: 65 | ARMOR CLASS: 15

Ettins, or two-headed Giants, are vicious and unpredictable hunters that stalk the night. Their two heads make them exceptionally sharp-eyed and alert.

STR	23/6	INT	6/-2	FOR	9
DEX	8/-1	WIS	10/0	REF	2
CON	15/2	CHA	11/0	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Alertness **Combat Feats:** Ambidexterity; Weapon Proficiency (simple); Two-Weapon Fighting **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Ettin Club; Spear

Fire Giant

RACE: Giant | ALIGNMENT: N

CHALLENGE RATING: 12 | HIT POINTS: 142 | ARMOR CLASS: 15

Brutal and ruthless, Fire Giants resemble huge, dark-skinned Dwarves. Well-organized and militaristic, they are typically encountered as well-armed phalanxes or twinned patrols.

STR	31/10	INT	10/0	FOR	14
DEX	9/-1	WIS	10/0	REF	4
CON	21/5	CHA	11/0	WIL	9

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** N/A **Combat Feats:** Cleave; Weapon Proficiency (martial) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Fire Giant / Hellhound Prop. (DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus) **Std. Eqp.:** Summoned Helmed Horror Sword; Chainmail

Frost Giant

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 10 | HIT POINTS: 133 | ARMOR CLASS: 16

Frost Giants are justifiably feared as brutal and wantonly destructive raiders. They have the appearence of oversized, muscular humans with snow-white or ivory skin.

STR	29/9	INT	10/0	SAVES FOR	14
DEX	9/-1	WIS	10/0	REF	3
CON	21/5	CHA	11/0	WIL	9

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Frost Giant / Winter Wolf Prop. (DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus) **Std. Eqp.:** War Hammer; Chain Shirt

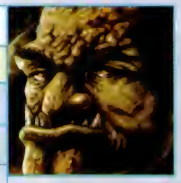
Hill Giant

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 9 | HIT POINTS: 102 | ARMOR CLASS: 17

Hunters and raiders with a selfish streak. Hill Giants seldom wash or repair their garments, preferring to add hides when their old ones wear out.

STR	25/7	INT	6/-2	SAVES FOR	12
DEX	8/-1	WIS	10/0	REF	3
CON	19/4	CHA	17/3	WIL	8



FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (simple); Weapon Focus (club) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Ogre/Hill Giant Club; Hide Armor

Mountain Giant

RACE: Giant | ALIGNMENT: CN

CHALLENGE RATING: 9 | HIT POINTS: 102 | ARMOR CLASS: 17

These brutish cave dwellers live communally and some have a basic grasp of magic. Intensely protective of their kin, they are generally reluctant to trust outsiders.

STR	25/7	INT	6/-2	SAVES FOR	12
DEX	8/-1	WIS	10/0	REF	3
CON	19/4	CHA	17/3	WIL	8

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (simple); Weapon Focus (club) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Ogre/Hill Giant Club; Hide Armor

GROUP Ogres

STRATEGY: Ogres are strong fighters that can be overwhelming in large numbers. Most are brainless warriors, but higher level encounters include talented Berserkers and versatile Mages. The mages should be your first target… They lack the incredible constitution of the fighters and should isolated and brought down quickly by the Fighters in your party before they do much damage. Ogres as a group are very vulnerable to area effect spells like Fireball and Web, and the powerful berserkers are reasonably easy to Charm or Dominate, a great way to even the odds.

SOCIETY: Scavenging, pillaging Ogres travel in small, poorly organized bands (2-8) looking for undefended settlements to raid or travelers to ambush. Intelligent Ogres may learn to speak common and seek alliances with other evil races or work as mercenaries. Ogre Mages are a slightly different species that live amidst other Ogres. They use their power and wits to lead their bands in more organized and purposeful ways, and often take slaves with their plunder.

Ogre (1)

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 13

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	6/-2	FOR	6
DEX	8/-1	WIS	10/0	REF	0
CON	15/2	CHA	7/-2	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (club) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Ogre / Hill Giant Club

Ogre (2)

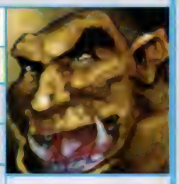
RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 13

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	6/-2	FOR	6
DEX	8/-1	WIS	10/0	REF	0
CON	15/2	CHA	7/-2	WIL	4

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (club) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Ogre/Hill Giant Club; Hide Armor

Ogre Berserker (1)

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 6 | HIT POINTS: 76 | ARMOR CLASS: 16

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	6/-2	FOR	12
DEX	14/2	WIS	10/0	REF	4
CON	18/4	CHA	7/-2	WIL	8

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Shield Proficiency **Other Feats:** Barbarian Fast Movement; Dark Vision; Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** BattleAxe; Chainmail

Ogre Berserker (2)

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 6 | HIT POINTS: 76 | ARMOR CLASS: 16

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	6/-2	FOR	12
DEX	14/2	WIS	10/0	REF	4
CON	18/4	CHA	7/-2	WIL	8

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Shield Proficiency **Other Feats:** Barbarian Fast Movement; Dark Vision; Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Longsword; Chainmail

Ogre Chieftain

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 9 | HIT POINTS: 150 | ARMOR CLASS: 14

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	6/-2	SAVES FOR	13
DEX	14/2	WIS	10/0	SAVES REF	5
CON	18/4	CHA	7/-2	SAVES WIL	9

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (exotic, simple); Wapon Focus (double axe) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Shield Proficiency **Other Feats:** Barbarian Fast Movement; Darkvision; Uncanny Dodge 1 **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Double Axe +1

Ogre High Mage

RACE: Giant | ALIGNMENT: LE

CHALLENGE RATING: 21 | HIT POINTS: 200 | ARMOR CLASS: 14

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	19/4	SAVES FOR	5
DEX	14/2	WIS	12/1	SAVES REF	6
CON	13/1	CHA	10/0	SAVES WIL	9

FEATS & ABILITIES

Active Combat Feats:N/A **Class/Racial Feats:** Alertness; Proficiency (creat.) **Combat Feats:** Weapon Proficiency (simple, wizard) **Defensive Feats:** Dodge **Magical Feats:** Combat Casting; Empower Spell; Quicken Spell; Spell Focus (Evocation) **Other Feats:** Darkvision; Summon Familiar **Monster Abilities:** 1 x Aura of Stunning **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Club +3

Ogre Mage (1)

RACE: Giant | ALIGNMENT: LE

CHALLENGE RATING: 5 | HIT POINTS: 37 | ARMOR CLASS: 14

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	14/2	SAVES FOR	7
DEX	10/0	WIS	14/2	SAVES REF	1
CON	17/3	CHA	17/3	SAVES WIL	6

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 9

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Greatsword +1; Chain Shirt

Ogre Mage (2)

RACE: Giant | ALIGNMENT: LE

CHALLENGE RATING: 5 | HIT POINTS: 37 | ARMOR CLASS: 14

Ogres are ugly, greedy creatures who live by raiding and scavenging. They associate freely with Giants and Trolls, banding together to prey on the weak.

STR	21/5	INT	14/2	SAVES FOR	7
DEX	10/0	WIS	14/2	SAVES REF	1
CON	17/3	CHA	17/3	SAVES WIL	6

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 9

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Greatsword +1; Chain Shirt

STRATEGY: Trolls no longer require a killing blow from a fire or acid weapon or spell, but they still regenerate quickly, making it necessary to isolate them and overwhelm them one-by-one. The Shaman's spells aren't particularly dangerous, put Troll Chieftains have some annoying spells, and should be eliminated immediately.

SOCIETY: Trolls are insatiable carnivores that can be a scourge to humanity in any climate. They tend to lack intelligence, have no governing structure, and are rarely found in groups of more than four.

GROUP: Trolls

Troll

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 63 | ARMOR CLASS: 18

Trolls are horrid carnivores, devouring everything from grubs to humanoids. Most creatures avoid these battles, as they know no fear and attack unceasingly when hungry.

STR	23/6	INT	6/-2	FOR	11
DEX	14/2	WIS	9/-1	REF	4
CON	23/6	CHA	6/-2	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Iron Will **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite1d6 **Skin/Hide:** Troll Prop. (Regeneration +2) **Std. Eqp.:** N/A

Troll Berserker

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 8 | HIT POINTS: 113 | ARMOR CLASS: 18

Trolls are horrid carnivores, devouring everything from grubs to humanoids. Most creatures avoid these battles, as they know no fear and attack unceasingly when hungry.

STR	23/6	INT	6/-2	FOR	15
DEX	14/2	WIS	9/-1	REF	5
CON	23/6	CHA	6/-2	WIL	8

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Prof. (light, medium); Dodge; Iron Will; Shield Proficiency **Other Feats:** Barbarian Fast Movement; Darkvision; Uncanny Dodge I **Monster Abilities:** 2x Rage **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite1d6 **Skin/Hide:** Troll Prop. (Regeneration +2) **Std. Eqp.:** N/A

Troll Chieftain

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 9 | HIT POINTS: 66 | ARMOR CLASS: 18

Trolls are horrid carnivores, devouring everything from grubs to humanoids. Most creatures avoid these battles, as they know no fear and attack unceasingly when hungry.

STR	23/6	INT	10/0	FOR	12
DEX	14/2	WIS	10/0	REF	6
CON	15/2	CHA	15/2	WIL	10

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Shield Proficiency **Magical Feats:** Combat Casting; Spell Focus (Evocation) **Other Feats:** Darkvision; Summon Familiar **Monster Abilities:** 1 x Aura of Fear **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite1d6 **Skin/Hide:** Troll Prop. (Regeneration +2) **Std. Eqp.:** N/A

Troll Shaman

RACE: Giant | ALIGNMENT: CE

CHALLENGE RATING: 7 | HIT POINTS: 105 | ARMOR CLASS: 18

Trolls are horrid carnivores, devouring everything from grubs to humanoids. Most creatures avoid these battles, as they know no fear and attack unceasingly when hungry.

STR	23/6	INT	9/-1	FOR	15
DEX	14/2	WIS	11/0	REF	5
CON	23/6	CHA	6/-2	WIL	9

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Iron Will **Magical Feats:** Combat Casting **Other Feats:** Darkvision; Turn Undead **Monster Abilities:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite1d6 **Skin/Hide:** Troll Prop. (Regeneration +2) **Std. Eqp.:** N/A

HUMANOIDS

GROUP Bugbears

STRATEGY: Bugbears are stronger, larger goblins and don't require any fancy strategy. They have low saving throws, so area-effect spells like Web and Confusion are effective. The Shamans are not particularly powerful, and the Hero and Chieftain are just stronger versions of regular Bugbears.

SOCIETY: Bugbears live in small tribes underground, where they organize simple raiding expeditions to get food and treasure. Occasionally Bugbears lead armies of Goblins, whom they bully into submission.

Bugbear (1)

RACE: Goblinoid | ALIGNMENT: CE

CHALLENGE RATING: 2 | HIT POINTS: 16 | ARMOR CLASS: 14

The biggest and strongest of the goblinoids. Bugbears receive their name from their bear-like noises, strong upper body, and keen sense of smell.

STR	15/2	INT	10/0	FOR	4
DEX	12/1	WIS	10/0	REF	2
CON	13/1	CHA	9/-1	WIL	1

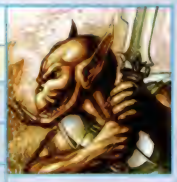
SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Morning Star; Leather Armor



Bugbear (2)

RACE: Goblinoid | ALIGNMENT: CE

CHALLENGE RATING: 2 | HIT POINTS: 16 | ARMOR CLASS: 14

The biggest and strongest of the goblinoids. Bugbears receive their name from their bear-like noises, strong upper body, and keen sense of smell.

STR	15/2	INT	10/0	FOR	4
DEX	12/1	WIS	10/0	REF	2
CON	13/1	CHA	9/-1	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Bolt; Light Crossbow; Leather Armor

Bugbear Chieftain

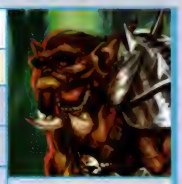
RACE: Goblinoid | ALIGNMENT: CE

CHALLENGE RATING: 10 | HIT POINTS: 150 | ARMOR CLASS: 17

The biggest and strongest of the goblinoids. Bugbears receive their name from their bear-like noises, strong upper body, and keen sense of smell.

STR	18/4	INT	10/0	FOR	10
DEX	12/1	WIS	10/0	REF	4
CON	13/1	CHA	12/1	WIL	3

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (exotic, simple) **Defensive Feats:** Armor Proficiency (light, medium); Great Fortitude **Other Feats:** Uncanny Dodge 1 **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Katana +1; Hide Armor

Bugbear Hero (1)

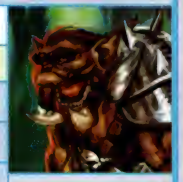
RACE: Goblinoid | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 36 | ARMOR CLASS: 17

The biggest and strongest of the goblinoids. Bugbears receive their name from their bear-like noises, strong upper body, and keen sense of smell.

STR	18/4	INT	10/0	FOR	7
DEX	12/1	WIS	10/0	REF	3
CON	13/1	CHA	12/1	WIL	2

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (bastard sword) **Defensive Feats:** Great Fortitude **Other Feats:** Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Bastard Sword +1; Hide Armor

Bugbear Hero (2)

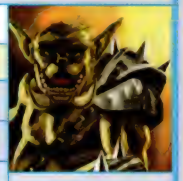
RACE: Goblinoid | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 36 | ARMOR CLASS: 17

The biggest and strongest of the goblinoids. Bugbears recieve their name from their bear-like noises, strong upper body, and keen sense of smell.

STR	18/4	INT	10/0	FOR	7
DEX	12/1	WIS	10/0	REF	3
CON	13/1	CHA	12/1	WIL	2

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (battleaxe) **Defensive Feats:** Great Fortitude **Other Feats:** Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Greataxe +1; Hide Armor

Bugbear Shaman (1)

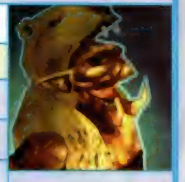
RACE: Goblinoid | ALIGNMENT: NE

CHALLENGE RATING: 4 | HIT POINTS: 33 | ARMOR CLASS: 14

The biggest and strongest of the goblinoids. Bugbears recieve their name from their bear-like noises, strong upper body, and keen sense of smell.

STR	15/2	INT	12/1	FOR	7
DEX	12/1	WIS	10/0	REF	3
CON	14/2	CHA	15/2	WIL	2

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** Combat Casting **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Sorcerer Level 1

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Dagger

Bugbear Shaman (2)

RACE: Goblinoid | ALIGNMENT: CE

CHALLENGE RATING: 4 | HIT POINTS: 33 | ARMOR CLASS: 14

The biggest and strongest of the goblinoids. Bugbears receive their name from their bear-like noises, strong upper body, and keen sense of smell.

				SAVES		
STR	15/2	INT	10/0		FOR	8
DEX	12/1	WIS	15/2		REF	3
CON	14/2	CHA	9/-1		WIL	4

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** Combat Casting **Other Feats:** Turn Undead **Monster Abilities:** N/A **Spells:** Cleric Level 2

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Mace; Leather Armor

GROUP: Fey

STRATEGY: In melee combat, Nixies, Pixies, and Grigs have little hope of prevailing– their hit points max out at 3. It's getting within melee range that's the trick: Nixies will try to Charm you, Pixies will try to Entangle or Confuse you, and Grigs will either Entangle you or disappear entirely. Long range weapons are effective, but not spells; all have considerable Spell Resistance.

Dryads and Nymphs are normal size, but not much tougher. They have similar spells, with the addition of the Nymph's blinding Aura of Unearthly Visage.

SOCIETY: Nymphs live solitary lives, and are usually very shy. They will attempt to heal any injured beasts that come seeking aid, even ones that could be potentially dangerous. They may help good humanoids as well, particularly ones that are unusually beautiful. Dryads are tree sprites, similarly shy and beautiful.

Of the three small sprites, Pixies and Grigs are Chaotic Good; they play pranks on unsuspecting travelers, but wish them no harm. They will fight to the death to protect their homelands from evil. Unlike the rest of the forest dwelling Fey, Nixies live in small ponds and lakes. They are very reclusive and may react with hostility towards outsiders.

Dryad

RACE: Fey | ALIGNMENT: LE

CHALLENGE RATING: 2 | HIT POINTS: 7 | ARMOR CLASS: 12

Dryads are exquisitely beautiful tree sprites found in secluded woodlands. They sometimes aid adventurers and can prove a source of useful information.

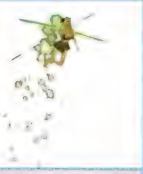
				SAVES		
STR	10/0	INT	14/2		FOR	0
DEX	15/2	WIS	15/2		REF	5
CON	11/0	CHA	18/4		WIL	5

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Dodge **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Caster Level 4

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Dagger



Grig

RACE: Fey | ALIGNMENT: NG

CHALLENGE RATING: 2 | HIT POINTS: 2 | ARMOR CLASS: 17

Mischievous and lighthearted, Grigs have no fear of big people and take great joy in playing tricks on them.

				SAVES		
STR	5/-3	INT	10/0		FOR	1
DEX	18/4	WIS	13/1		REF	6
CON	13/1	CHA	14/2		WIL	3

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Caster Level 9

EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** Bite1d2 **Special Attack:** Gore1d2 **Skin/Hide:** Grig/Nixie/Pixie Prop. (Spell Resistance Bonus of 16 SR) **Std. Eqp.:** N/A

Nixie

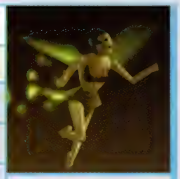
RACE: Fey | ALIGNMENT: N

CHALLENGE RATING: 1 | HIT POINTS: 3 | ARMOR CLASS: 14

With their webbed fingers and toes, the tiny Nixies are well suited to their amphibious lifesyle. It's rare to find one far from its lake-shore home.

STR	7/-2	INT	12/1	FOR	0
DEX	16/3	WIS	13/1	REF	5
CON	11/0	CHA	18/4	WIL	3

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Other Feats:** N/A **Defensive Feats:** Dodge **Monster Abilities:** N/A **Spells:** Caster Level 4

EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** Slam1d2 **Special Attack:** Gore1d2 **Skin/Hide:** Grig/Nixie/Pixie Prop. (Spell Resistance Bonus of 16 SR) **Std. Eqp.:** Dagger

Nymph

RACE: Fey | ALIGNMENT: LE

CHALLENGE RATING: 3 | HIT POINTS: 10 | ARMOR CLASS: 11

Nymphs are nature's embodiment of physical beauty: They are so unbearably lovely that even a glimpse can blind or kill unwary lookers.

STR	10/0	INT	16/3	FOR	1
DEX	13/1	WIS	17/3	REF	4
CON	10/0	CHA	19/4	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge; Iron Will **Other Feats:** N/A **Monster Abilities:** 1 X Aura of Unearthly Visage **Spells:** Caster Level 7

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Dagger

Pixie

RACE: Fey | ALIGNMENT: NG

CHALLENGE RATING: 2 | HIT POINTS: 3 | ARMOR CLASS: 16

Talented illusionists and irreverent pranksters, Pixies make their homes in hollowed trees and rocky grottoes, deep within forests.

STR	7/-2	INT	16/3	FOR	0
DEX	18/4	WIS	15/2	REF	6
CON	11/0	CHA	16/3	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Point Blank Shot; Weapon Finesse **Defensive Feats:** Dodge **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Caster Level 8

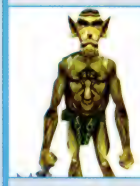
EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** Slam1d2 **Special Attack:** Gore1d2 **Skin/Hide:** Grig/Nixie/Pixie Prop. (Spell Resistance Bonus of 16 SR) **Std. Eqp.:** N/A

GROUP Goblins

STRATEGY: Goblins have no special properties and are one of the easiest encounters in the game. The only real danger they pose is the phenomenal numbers in which they may attack. The spellcasting abilities of the Shamans are not significant, but the Chieftain can use spells up to level 4 and should be hounded by constant attacks to fizzle potentially dangerous spells.

SOCIETY: Goblins live in tribes that may number into the hundreds. They live in dank caves, and tend to make their homes near civilized races from whom they can steal, pillage, and enslave workers. They may form alliances with Bugbears or Worgs.



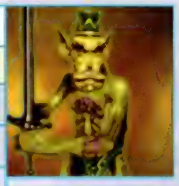
Goblin (1)

RACE: Goblinoid | ALIGNMENT: NE

CHALLENGE RATING: 1/4 | HIT POINTS: 4 | ARMOR CLASS: 10

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

				SAVES		
STR	8/-1	INT	10/0		FOR	2
DEX	8/-1	WIS	11/0		REF	0
CON	11/0	CHA	8/-1		WIL	0

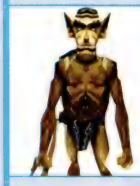


FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Morningstar



Goblin (2)

RACE: Goblinoid | ALIGNMENT: CN

CHALLENGE RATING: 1/4 | HIT POINTS: 4 | ARMOR CLASS: 10

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

				SAVES		
STR	8/-1	INT	10/0		FOR	2
DEX	9/-1	WIS	11/0		REF	0
CON	11/0	CHA	8/-1		WIL	0

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial) **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Arrow; Shortbow

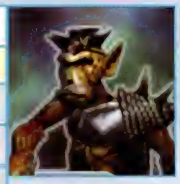
Goblin Chieftain

RACE: Goblinoid | ALIGNMENT: NE

CHALLENGE RATING: 11 | HIT POINTS: 90 | ARMOR CLASS: 14

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

				SAVES		
STR	10/0	INT	10/0		FOR	4
DEX	14/2	WIS	12/1		REF	5
CON	13/1	CHA	17/3		WIL	7



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Proficiency (simple); Weapon Finesse **Defensive Feats:** Shield Prof. **Magical Feats:** Combat Casting; Spell Focus (evocation) **Other Feats:** Summon Familiar **Monster Abilities:** 1 X Aura of Electricity **Spells:** Sorcerer Level 4

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Dagger

Goblin Elite (1)

RACE: Goblinoid | ALIGNMENT: NE

CHALLENGE RATING: 4 | HIT POINTS: 18 | ARMOR CLASS: 17

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

STR	12/1	INT	10/0	FOR	6
DEX	16/3	WIS	11/0	REF	4
CON	11/0	CHA	8/-1	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Profiency (creature) **Combat Feats:** Weapon Proficiency (rogue, simple); Two-Weapon Fighting **Defensive Feats:** Armor Proficiency (light) **Magical Feats:** N/A **Other Feats:** Evasion; Sneak Attack (+1d6); Uncanny Dodge 1 **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Short Sword; Studded Leather Armor; Dagger

Goblin Elite (2)

RACE: Goblinoid | ALIGNMENT: CN

CHALLENGE RATING: 3 | HIT POINTS: 17 | ARMOR CLASS: 18

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

STR	15/2	INT	10/0	FOR	5
DEX	13/1	WIS	11/0	REF	1
CON	11/0	CHA	8/-1	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Shield Proficiency **Magical Feats:** N/A **Other Feats:** Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Handaxe; Studded Leather Armor; Small Shield

Goblin Shaman (1)

RACE: Goblinoid | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 18 | ARMOR CLASS: 14

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

STR	8/-1	INT	10/0	FOR	5
DEX	13/1	WIS	15/2	REF	2
CON	11/0	CHA	8/-1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency **Magical Feats:** Combat Casting **Other Feats:** Turn Undead **Monster Abilities:** N/A **Spells:** Cleric Level 2

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Warhammer; Chain Shirt

Goblin Shaman (2)

RACE: Goblinoid | ALIGNMENT: CN

CHALLENGE RATING: 3 | HIT POINTS: 12 | ARMOR CLASS: 18

While Goblins are often considered little more than a nuisance, an unchecked population can often balloon into a serious threat to normally civilized areas.

STR	8/-1	INT	10/0	FOR	5
DEX	13/1	WIS	11/0	REF	2
CON	11/0	CHA	15/2	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Prof. (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** Combat Casting; Spell Focus (Evocation) **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Sorcerer Level 1

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Dagger

GROUP
Lizardfolk

STRATEGY: The Lizardfolk are another race that are dangerous only to low-level characters. Their warriors are traditional fighters with no exemplary abilities, their shamans have only level 1 and 2 spells, and their chieftains are not particularly powerful (but the +2d6 Sneak Attack ability one of them has can be a nasty surprise).

The Yuan-Ti are another matter entirely. Even the warriors have a handful of nasty spell-like abilities like Entangle and Darkness, and the Sorcerer means business with Fireball in all three third level spell slots. A Spell Resistance of 16 ensures that all but the most powerful mages stand a high risk of wasting their offensive spells, so you'll need to rely on constant, powerful physical attacks. Fortunately, they're not so tough when in melee range.

SOCIETY: Lizardfolk and Yuan-Ti are entirely different species that do not intermingle. Lizardfolk are amphibious, marsh-dwelling omnivores that are not inherently evil but do enjoy eating human flesh. They have a primitive tribal structure, and subsist primarily on fishing and scavenging. Yuan-Ti are part-snake Humanoids known for their cruelty, cunning, and brilliance. They are deeply religious worshippers of evil whose society revolves around temples hidden in forests or underground.

Lizardfolk Chieftain (1)

RACE: Reptilian | ALIGNMENT: NE

CHALLENGE RATING: 4 | HIT POINTS: 38 | ARMOR CLASS: 16

Primitive and savage, these reptilian humanoids can be very dangerous if provoked. They are fiercely territorial, banding together to protect the swamplands they inhabit.

STR	15/2	INT	9/-1	FOR	8
DEX	12/1	WIS	10/0	REF	2
CON	15/2	CHA	10/0	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Shield Proficiency **Other Feats:** Barbarian Fast Movement, Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/ **Std. Eqp.:** BattleAxe

Lizardfolk Chieftain (2)

RACE: Reptilian | ALIGNMENT: NE

CHALLENGE RATING: 4 | HIT POINTS: 24 | ARMOR CLASS: 17

Primitive and savage, these reptilian humanoids can be very dangerous if provoked. They are fiercely territorial, banding together to protect the swamplands they inhabit.

STR	13/1	INT	9/-1	FOR	7
DEX	15/2	WIS	10/0	REF	3
CON	13/1	CHA	10/0	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency (rogue, simple); Weapon Finesse; Two-Weapon Fighting **Defensive Feats:** Armor Proficiency (light) **Other Feats:** Evasion; Sneak Attack (+1d6, +2d6); Uncanny Dodge 1 **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Rapier; Rapier

Lizardfolk Shaman (1)

RACE: Reptilian | ALIGNMENT: N

CHALLENGE RATING: 4 | HIT POINTS: 16 | ARMOR CLASS: 16

Primitive and savage, these reptilian humanoids can be very dangerous if provoked. They are fiercely territorial, banding together to protect the swamplands they inhabit.

STR	10/0	INT	10/0	FOR	8
DEX	13/1	WIS	12/1	REF	2
CON	14/2	CHA	15/2	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** Combat Casting **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Sorcerer Level 1

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d4 **Skin/Hide:** N/A **Std. Eqp.:** Quarterstaff

Lizardfolk Shaman (2)

RACE: Reptilian | ALIGNMENT: NE

CHALLENGE RATING: 4 | HIT POINTS: 27 | ARMOR CLASS: 15

Primitive and savage, these reptilian humanoids can be very dangerous if provoked. They are fiercely territorial, banding together to protect the swamplands they inhabit.

STR	13/1	INT	9/-1	FOR	8
DEX	10/0	WIS	15/2	REF	1
CON	14/2	CHA	10/0	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency **Magical Feats:** Combat Casting **Other Feats:** Turn Undead **Monster Abilities:** N/A **Spells:** Cleric Level 2

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Std. Eqp.:** Club

Lizardfolk Warrior (1)

RACE: Reptilian | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 11 | ARMOR CLASS: 15

Primitive and savage, these reptilian humanoids can be very dangerous if provoked. They are fiercely territorial, banding together to protect the swamplands they inhabit.

STR	13/1	INT	9/-1	FOR	4
DEX	10/0	WIS	10/0	REF	0
CON	13/1	CHA	10/0	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Prof. (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Bolt; Heavy Crossbow

Lizardfolk Warrior (2)

RACE: Reptilian | ALIGNMENT: N

CHALLENGE RATING: 2 | HIT POINTS: 11 | ARMOR CLASS: 15

Primitive and savage, these reptilian Humanoids can be very dangerous if provoked. They are fiercely territorial, banding together to protect the swamplands they inhabit.

STR	13/1	INT	9/-1	FOR	4
DEX	10/0	WIS	10/0	REF	0
CON	13/1	CHA	10/0	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Prof. (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Scimitar

Yuan-Ti

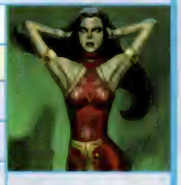
RACE: Reptilian | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 42 | ARMOR CLASS: 12

Devout worshipers of evil, pure-blooded Yuan-Ti often appear Human at first glance. It is only when they draw close that their snake-like features can be seen.

STR	11/0	INT	18/4	FOR	5
DEX	13/1	WIS	18/4	REF	3
CON	11/0	CHA	16/3	WIL	6

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (martial, simple) **Defensive Feats:** Dodge **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Caster Level 8

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Yuan-Ti Prop. (Spell Resistance Bonus of 16 SR) **Std. Eqp.:** Dagger

Yuan-Ti Priest

RACE: Reptilian | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 42 | ARMOR CLASS: 12

Devout worshipers of evil, pure-blooded Yuan-Ti often appear Human at first glance. It is only when they draw close that their snake-like features can be seen.

				SAVES	
STR	13/1	INT	18/4	FOR	7
DEX	13/1	WIS	18/4	REF	3
CON	14/2	CHA	16/3	WIL	9

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (mace) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Dodge; Shield Proficiency **Magical Feats:** Combat Casting **Other Feats:** Turn Undead **Monster Abilities:** N/A **Spells:** Cleric Level 3

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Mace; Leather Armor

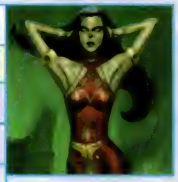
Yuan-Ti Sorcerer

RACE: Reptilian | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 28 | ARMOR CLASS: 12

Devout worshipers of evil, pure-blooded Yuan-Ti often appear Human at first glance. It is only when they draw close that their snake-like features can be seen.

				SAVES	
STR	11/0	INT	18/4	FOR	4
DEX	13/1	WIS	18/4	REF	3
CON	14/2	CHA	16/3	WIL	9



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Dodge; Shield Proficiency **Magical Feats:** Combat Casting; Spell Focus (Evocation) **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** _____

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Yuan-Ti Prop. (Spell Resistance Bonus of 16 SR) **Std. Eqp.:** Dagger

GROUP: Minotaurs

STRATEGY: Minotaurs are very talented warriors that come from the the-best-defense-is-dismembering-your-opponent-with-an-ax school of fighting. As such, they dish out a serious beating, but have unimpressive Armor Classes and little ability to resist offensive spells that require a Reflex save (their Fortitude and Will saves are impressive, however). The Shamans have a few divine spells but nothing that isn't preferable to being hit with a Greataxe. The Chieftain has the ability to throw fire bolts, but with 250 HP and incredible fortitude, you won't be able to stop him. Instead, focus your attacks on the weakest member and work your way up to the tougher ones.

SOCIETY: Minotaurs are not particularly bright, but have an innate ability never to become lost. This is why they tend to make their homes and hunting grounds out of complex underground labyrinths. They hunt alone or in packs of up to 4, and enjoy letting their complex and dangerous homes wear down intruders before they go in for the kill.

Minotaur

RACE: Monstrous | ALIGNMENT: CE

CHALLENGE RATING: 4 | HIT POINTS: 39 | ARMOR CLASS: 14

Strong, fiercely territorial creatures, Minotaurs often lair in vast underground labyrinths where their keen memory gives them an advantage over their lost and confused prey.

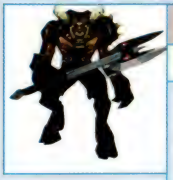
				SAVES	
STR	19/4	INT	7/-2	FOR	7
DEX	10/0	WIS	10/0	REF	5
CON	15/2	CHA	8/-1	WIL	2

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial); Weapon Specialization (battleaxe) **Defensive Feats:** Great Fortitude **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Gore1d8 **Skin/Hide:** N/A **Std. Eqp.:** Minotaur / Minogon / Frost Giant Greataxe



Minotaur Berserker

RACE: Monstrous | ALIGNMENT: CE

CHALLENGE RATING: 8 | HIT POINTS: 90 | ARMOR CLASS: 14

Strong, fiercely territorial creatures, Minotaurs often lair in vast underground labyrinths where their keen memory gives them an advantage over their lost and confused prey.

STR	20/5	INT	7/-2	FOR	12
DEX	10/0	WIS	10/0	REF	10
CON	15/2	CHA	8/-1	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Great Fortitude **Other Feats:** Barbarian Fast Movement; Uncanny Dodge 1, 2 **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Std. Eqp.:** Greataxe +1

Minotaur Chieftain

RACE: Monstrous | ALIGNMENT: CE

CHALLENGE RATING: 14 | HIT POINTS: 250 | ARMOR CLASS: 14

Strong, fiercely territorial creatures, Minotaurs often lair in vast underground labyrinths where their keen memory gives them an advantage over their lost and confused prey.

STR	19/4	INT	7/-2	FOR	12
DEX	10/0	WIS	10/0	REF	10
CON	15/2	CHA	8/-1	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Prof.(creature) **Combat Feats:** Cleave; Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Great Fortitude **Other Feats:** Barbarian Fast Movement; Uncanny Dodge 1, 2 **Monster Abilities:** 2 x Bolt, Fire **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Std. Eqp.:** Minotaur Chieftain Greataxe

Minotaur Shaman

RACE: Monstrous | ALIGNMENT: CE

CHALLENGE RATING: 9 | HIT POINTS: 82 | ARMOR CLASS: 14

Strong, fiercely territorial creatures, Minotaurs often lair in vast underground labyrinths where their keen memory gives them an advantage over their lost and confused prey.

STR	19/4	INT	9/-1	FOR	12
DEX	10/0	WIS	15/2	REF	10
CON	14/2	CHA	8/-1	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature); Skill Focus (Spellcraft) **Combat Feats:** Proficiency (martial, simple) **Defensive Feats:** N/A **Magical Feats:** Combat Casting **Other Feats:** Turn Undead **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Gore1d8 **Skin/Hide:** N/A **Std. Eqp.:** Minotaur / Minogon / Frost Giant Greataxe

GROUP Orcs

STRATEGY: Orcs are yet another species of warriors with few interesting characteristics. Their greatest strength is in numbers, and an Orc battalion includes an interesting variety of Barbarians, Clerics, Sorcerers and even Rogues.

SOCIETY: Orcs are the primary threat to human, elf, and dwarf settlements in many parts of Faerûn. Their lives revolve around battle and their society is exclusively militant. Their drive to conquer as much territory as possible puts them at odds with all other intelligent beings, above the ground and below.

Orc (1)

RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 1/4 | HIT POINTS: 4 | ARMOR CLASS: 10

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	15/2	INT	9/-1	SAVES FOR	2
DEX	10/0	WIS	8/-1	REF	0
CON	11/0	CHA	8/-1	WIL	0

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** N/A **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Greataxe

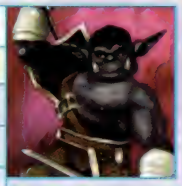
Orc (2)

RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 1/4 | HIT POINTS: 4 | ARMOR CLASS: 10

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	15/2	INT	9/-1	SAVES FOR	2
DEX	10/0	WIS	8/-1	REF	0
CON	11/0	CHA	8/-1	WIL	0



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial) **Defensive Feats:** N/A **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Arrow; Longbow

Orc Champion (1)

RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 24 | ARMOR CLASS: 10

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	15/2	INT	9/-1	SAVES FOR	5
DEX	10/0	WIS	8/-1	REF	1
CON	11/0	CHA	8/-1	WIL	0

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Iron Will **Magical Feats:** N/A **Other Feats:** Barbarian Fast Movement; Uncanny Dodge 1 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Greataxe; Scale Mail

Orc Champion (2)

RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 18 | ARMOR CLASS: 11

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	15/2	INT	9/-1	FOR	6
DEX	11/0	WIS	8/-1	REF	1
CON	10/0	CHA	8/-1	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Prof. (creature) **Combat Feats:** Cleave; Weapon Proficiency (rogue, simple) **Defensive Feats:** Armor Proficiency (light); Dodge **Magical Feats:** N/A **Other Feats:** Evasion; Sneak Attack (+1d6); Uncanny Dodge 1 **Monster Abilities:** 1 x Intensity **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Morningstar; Studded Leather Armor

Orc Chieftain

RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 10 | HIT POINTS: 120 | ARMOR CLASS: 14

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	12/1	INT	9/-1	FOR	7
DEX	11/0	WIS	8/-1	REF	3
CON	10/0	CHA	8/-1	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (exotic, simple) **Defensive Feats:** Armor Proficiency (light, medium); Dodge **Magical Feats:** N/A **Other Feats:** N/A **Monster Abilities:** 1 x Aura of Cold **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Two-Bladed Sword

Orc Shaman (1)

RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 4 | HIT POINTS: 32 | ARMOR CLASS: 10

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	15/2	INT	9/-1	FOR	8
DEX	10/0	WIS	15/2	REF	1
CON	14/2	CHA	8/-1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Armor Proficiency (light, medium); Shield Proficiency **Magical Feats:** Combat Casting **Other Feats:** Turn Undead **Monster Abilities:** N/A **Spells:** Cleric Level 2

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Quarterstaff; Hide Armor

Orc Shaman (2)

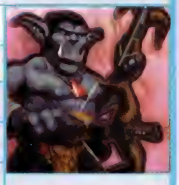
RACE: Orc | ALIGNMENT: CE

CHALLENGE RATING: 4 | HIT POINTS: 24 | ARMOR CLASS: 10

Orcs are aggressive Humanoids that raid, pillage, and battle other creatures. They have a hatred of Elves and Dwarves that goes back many generations.

STR	8/-1	INT	12/1	FOR	8
DEX	10/0	WIS	12/1	REF	1
CON	14/2	CHA	15/2	WIL	2

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Iron Will **Magical Feats:** Combat Casting **Other Feats:** N/A **Monster Abilities:** N/A **Spells:** Sorcerer Level 2

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** Quarterstaff

INSECTS

GROUP Beetles

STRATEGY: Beetles can be difficult encounters for low-level characters but should be simple for more powerful parties. Fire Beetles have little going for them other than numerical superiority; they should go down in a single hit from even a low-level character. Stag Beetles are much tougher, doing 4d6+6 damage per bite and sporting a hard-to-penetrate AC of 20. Hive Mothers have basically the same attributes as Stag Beetles, but with a few hundred more HP.

Stink Beetles can be dangerous for a low-level party, with their one-shot Stinking Cloud ability. Keep an eye out for weak party members that have been affected by the gas, as they could quickly be swarmed and devoured by the beetles. Bombardier Beetles have a nasty Cone of Acid, so keep some distance between your party members so they can't hit the whole group.

SOCIETY: Beetles are always neutral and will usually only attack when threatened. They eat crops and carrion, not live prey, but will attack those who defend a desired food source. Stag Beetles, particularly, are a scourge in rural areas.

Bombardier Beetle

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 16

Called such for their ability to lob balls of digestive acid over considerable distances, these giant Beetles feed primarily on carrion and offal.

STR	13/1	INT	3/-4	FOR	5
DEX	10/0	WIS	10/0	REF	0
CON	14/2	CHA	9/-1	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 2 x Cone of Acid **Spells:** N/A

EQUIPMENT

Claw 1: Bombardier/Stink Beetle Bite (2d4 Dmg.; On Monster Hit: Disease Vermin Madness) **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Vermin Prop. (Immunity: Misc.: Mind-Affecting Spells) **Std. Eqp.:** N/A

Fire Beetle

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 1/4 | HIT POINTS: 4 | ARMOR CLASS: 10

Recognizable by its distinctive red markings, the predatory Fire Beetle prefers its meals to be cooked, even a little burnt...

STR	10/0	INT	3/-4	FOR	2
DEX	11/0	WIS	10/0	REF	0
CON	11/0	CHA	7/-2	WIL	0

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Bite1d4 **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Vermin Prop. (Immunity: Misc.: Mind-Affecting Spells) **Std. Eqp.:** N/A

Hive Mother

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 15 | HIT POINTS: 262 | ARMOR CLASS: 16

These creatures are serious pests that greedily devour cultivated crops. Though not carnivorous, they'll still charge any perceived threat with their oversized, horn-like mandibles.

STR	23/6	INT	3/-4	FOR	10
DEX	10/0	WIS	10/0	REF	3
CON	17/3	CHA	9/-1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Stage Beetle / Hive Mother Bite (4d6 Dmg.; On Monster Hit: Disease Vermin Madness) **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Vermin Prop. (Immunity: Misc.: Mind-Affecting Spells) **Std. Eqp.:** N/A

Stag Beetle

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 5 | HIT POINTS: 52 | ARMOR CLASS: 20

These creatures are serious pests that greedily devour cultivated crops. Though not carnivorous, they'll still charge any percieved threat with their oversized, horn-like mandibles.

STR	23/6	INT	3/-4	FOR	8
DEX	10/0	WIS	10/0	REF	2
CON	17/3	CHA	9/-1	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Stag Beetle / Hive Mother Bite (4d6 Dmg.; On Monster Hit: Disease Vermin Madness) **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Vermin Prop. (Immunity: Misc.: Mind-Affecting Spells) **Std. Eqp.:** N/A

Stink Beetle

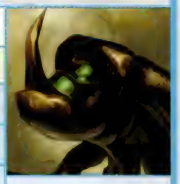
RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 19 | ARMOR CLASS: 16

As their name suggests, Stink Beetles defend themselves by emitting a noxious gas intended to render their assailants unconscious.

STR	14/2	INT	3/-4	FOR	5
DEX	10/0	WIS	10/0	REF	1
CON	15/2	CHA	6/-2	WIL	1

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 3

EQUIPMENT

Claw 1: Bombardier/Stink Beetle Bite (2d4 Dmg.; On Monster Hit: Disease Vermin Madness) **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Stink Beetle Prop. (Immunity: Misc.: Mind-Affecting Spells; Immunity: Specific Spell: Stinking Cloud) **Std. Eqp.:** N/A

GROUP: Spiders

STRATEGY: All Spiders have the ability to poison their victims, so anti-poison potions, spells, and scrolls will be welcome, but not essential. Remember that if afflicted, you take poison damage twice; once when hit, and once ten turns later. So there usually isn't much point tending to poison during battle, freeing up you to kill the Spider first. If you can't neutralize the poison later, don't worry–the effects (a loss of a certain ability score) will go away when you rest.
Phase Spiders are interesting enemies… They have a custom script that teleports them to the weakest (most damaged) enemy at the beginning of every round. Obviously, this can be very dangerous for characters that are low on health, so always keep your life bar out of the danger range.
Queen Spiders have Greater Stoneskin, which will gift them with 20/+5 damage reduction if it resolves successfully, and Globe of Invulnerability which will make them immune to spells of Level 4 or lower. This will make life very difficult if you don't have Lesser Spell Breach or some other way of dealing with it. All Spiders (except Phase Spiders) are immune to mind-affecting spells, and Wraith Spiders are also immune to cold and paralysis, and have Damage Reduction 10/+1.

SOCIETY: Phase Spiders dwell on both the Ethereal and Material Plane, and move between them when they hunt. That's about the most interesting thing I can come up with here. They're Spiders… What do you expect?

Dire Spider

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 7 | HIT POINTS: 55 | ARMOR CLASS: 17

Though not true Dire creatures, these Spiders are nevertheless fearsome to behold and appear to act with more cunning than their more common cousins.

STR	19/4	INT	3/-4	FOR	8
DEX	17/3	WIS	10/0	REF	6
CON	12/1	CHA	3/-4	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Dire Spider Bite (2d6 Dmg.; On Monster Hit: Poison Huge Spider Venom) **Claw 2:** Dire Spider Bite **Special Attack:** Dire Spider Bite **Skin/Hide:** Spider Prop. (Immunity: Misc.: Mind-Affecting Spells; Immunity: Specific Spell Web) **Std. Eqp.:** N/A

Giant Spider

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 22 | ARMOR CLASS: 14

Aggressive, nightmarish predators, Giant Spiders first trap their prey within their webs, then subdue them with poisonous venom.

				SAVES	
STR	15/2	INT	10/0	FOR	5
DEX	17/3	WIS	10/0	REF	4
CON	12/1	CHA	3/-4	WIL	1

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Sword/Giant Spider Bite (1d8 Dmg.; On Monster Hit: Poison Giant Spider Venom) **Claw 2:** Sword/Giant Spider Bite **Special Attack:** Sword/Giant Spider Bite **Skin/Hide:** Spider Prop. (Immunity: Misc.: Mind-Affecting Spells; Immunity: Specific Spell Web) **Std. Eqp.:** N/A

Phase Spider

RACE: M. Beast | ALIGNMENT: N

CHALLENGE RATING: 4 | HIT POINTS: 37 | ARMOR CLASS: 15

Aggressive predators that can "phase" back and forth between the Material and Ethereal Planes, the attacks of these magical creatures can be very disconcerting.

				SAVES	
STR	17/3	INT	7/-2	FOR	7
DEX	17/3	WIS	13/1	REF	4
CON	16/3	CHA	10/0	WIL	2

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Phase Spider Bite (1d6 Dmg.; On Monster Hit: Poison Phase Spider Venom) **Claw 2:** Phase Spider Bite **Special Attack:** Phase Spider Bite **Skin/Hide:** N/A **Std. Eqp.:** N/A

Queen Spider

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 13 | HIT POINTS: 250 | ARMOR CLASS: 16

Though not true Dire creatures, these Spiders are nevertheless fearsome to behold and appear to act with more cunning than their more common cousins.

				SAVES	
STR	19/4	INT	3/-4	FOR	8
DEX	17/3	WIS	10/0	REF	6
CON	12/1	CHA	3/-4	WIL	3

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Bolt, Web **Spells:** Caster Level 11

EQUIPMENT

Claw 1: Dire Spider Bite (2d6 Dmg.; On Monster Hit: Queen Spider Venom) **Claw 2:** Dire Spider Bite **Special Attack:** Dire Spider Bite **Skin/Hide:** Spider Prop. (Immunity: Misc.: Mind-Affecting Spells; Immunity: Specific Spell Web) **Std. Eqp.:** N/A

Sword Spider

RACE: Vermin | ALIGNMENT: N

CHALLENGE RATING: 4 | HIT POINTS: 32 | ARMOR CLASS: 17

With their chitinous, seemingly metallic shell, Sword Spiders are hard to kill. More dangerous, however, are the sharpened, blade-like appendages that give them their name.

				SAVES	
STR	18/4	INT	3/-4	FOR	6
DEX	10/0	WIS	11/0	REF	1
CON	14/2	CHA	3/-4	WIL	1

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Sword/Giant Spider Bite (1d8 Dmg.; On Monster Hit: Poison Large Spider Venom) **Claw 2:** Sword/Giant Spider Bite **Special Attack:** Sword/Giant Spider Bite **Skin/Hide:** Spider Prop. (Immunity: Misc.: Mind-Affecting Spells; Immunity: Specific Spell Web) **Std. Eqp.:** N/A

Wraith Spider

RACE: Undead | ALIGNMENT: CG

CHALLENGE RATING: 4 | HIT POINTS: 16 | ARMOR CLASS: 22

Permeated with the stench of decay, these undead vermin are amongst the most dangerous of the arachnids. Steer well clear of their chill touch.

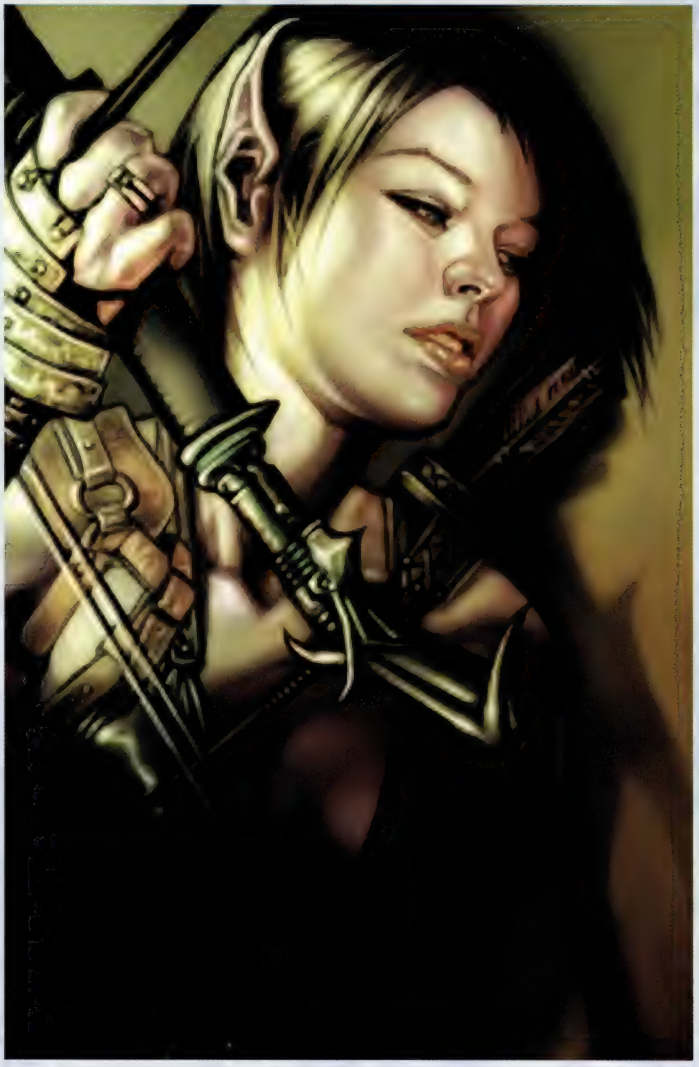
STR	11/0	INT	3/-4	SAVES FOR	4
DEX	17/3	WIS	10/0	REF	4
CON	12/1	CHA	3/-4	WIL	1

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Wraith Spider Bite (1d4 Dmg.; On Monster Hit: Level Drain Amount: 1; On Monster Hit: Poison Wraith Spider Venom) **Claw 2:** Wraith Spider Bite **Special Attack:** Wraith Spider Bite **Skin/Hide:** Wraith Spider Prop. (Immunity: Dmg. Type: Cold 100% Imm. Bonus; Immunity: Misc.: Mind-Affecting Spells, Paralysis) **Std. Eqp.:** N/A



MISCELLANEOUS

BATTLE/INTELLECT DEVOURER

STRATEGY: The Intellect Devourer you met in the official campaign had some fancy programming that these lack. The off-the-shelf Intellect Devourers are, perhaps, even more dangerous. With the ability to Dominate Person six times a day, Invisibility, and Damage Reduction 20/+3, Intellect Devourers can be brutal to small parties. Spellcasters can keep them on the ropes with offensive spells (stay away from fire and electricity), but fighters without +3 weapons won't have a lot of options. Savage Battle Devourers lack the ability to Dominate but more than compensate for that with an immunity to every elemental spell type, a Fear Gaze, and a Slow effect whenever they hit their prey. Tank characters and summoned beasts will be essential to hold them off, but don't expect them to get much done between the Slow effect and their obscene AC of 30. It will be up to the spellcasters to take them down with mind-control or death spells.
SOCIETY: Intellect Devourers were bred by Mind Flayers, a race of brilliant and evil beings that seek to rule the world with their psionic abilities. They live deep underground and prefer to stay safe while dominated slaves and Intellect Devourers do their bidding. Battle Devourers are a more powerful version used when the subtler nuances of the Intellect Devourers just won't cut it.

ETTERCAP

STRATEGY: Ettercaps utilize the same tactics and abilities as the spiders they often fight beside. They like to gum a party up in a big Web spell and then poison them with their painful bite. The venom lowers Dexterity, making victims more vulnerable to Webs, and so on. Parties going against Ettercaps and spiders should keep a good distance from each other, so when a web is fired the unaffected members can rush in and smash the weak-on-defense Ettercap.
SOCIETY: Ettercaps are cowardly but cunning. They set traps on heavily traveled trails and roads, especially in forests, and eat the trapped victims alive.

GARGOYLE

STRATEGY: Gargoyles think they're so clever, pretending to be Gargoyle-shaped statues until players wander by. But how often do you see Gargoyle-shaped statues when there aren't Gargoyles? It's rarely a surprise the second time. Fortunately for them, Gargoyles have more going for them than that. A good AC, a pile of hit points and Damage Reduction 15/+1 can be a problem for unprepared adventurers. Use enchanted weapons and Will-based spells.
SOCIETY: Gargoyles are vicious bullies that enjoy torturing their victims for the sheer thrill of it. They require no food and water and have infinite patience.

GRAY RENDER

STRATEGY: Gray Renders are simple beasts, with a ton of HP and two claws and a bite worth of beatings. Nothing fancy here… Kill them before they kill you.
SOCIETY: Gray Renders are deadly predators that never hunt with other Gray Renders. Instead, they may foster and protect a pack of lesser creatures, like wolves, lions, or even humanoids. They are very stupid, but kind of cute. They are the author of this book's favorite monster and he demands you use them respectfully, or not at all.

HOOK HORROR

STRATEGY: Hook Horrors are savage, quick, underground predators. They're quick on their feet and dish out a healthy amount of damage with their three attacks, which include a 2d12 bite. Let your expendable creatures and henchmen take the hits while you hammer them, preferably with powerful spells.

KRENSHAR

STRATEGY: If Krenshars were as fierce as they looked, we'd all be in trouble. Their ability to terrify away members of your party is a major annoyance, but they only do 1d2 damage a strike, so you'll probably still be alive when the frightened members come to their senses and make it back to the battlefield.
SOCIETY: Krenshars hunt in well organized packs of 6-10. A single scout then attempts to scare the rest of the party into fleeing towards the rest of the pack.

STIRGE

STRATEGY: Stirges are a threat only in large numbers, and have no particular resistances. That means even low-level area effect spells like Fireball, Sound Burst, and Stinking Cloud are useful. If that's not an option, then fighters should get a weapon in each hand (if you can), archers should switch to Rapid Shot, and spellcasters should start summoning. With only 5 HP, every hit is a kill, so the more attacks the better. Every successful Stirge attack drains a Constitution point, so a badly hit party will need Lesser Restoration spells or potions ready.
SOCIETY: Stirges hunt warm forests and caves in flocks of 3 to 9 looking for blood to suck. That's about all there is to it.

UMBER HULK

STRATEGY: Umber Hulks will try to Confuse their opponents and then beat the tar out of them while incapacitated. If you can, cast Magic Circle Against Alignment first or do whatever else you can to increase your Will Saving Throw. Hulks do a ton of damage, so wizards will want Ghostly Visage or Stoneskin or something to prevent them from being cut down in a single turn. Lots of Summoned Creatures will also lower the odds they attack you, while you whittle them down. Finally, Umber Hulks have lousy saving throws, so all spells are great.
SOCIETY: Umber Hulks live deep underground where they use their incredible strength to dig massive tunnels. They're smarter than they look, and will set traps and ambushes for prey.

WILL-O'-WISP

STRATEGY: Will-O'-Wisps are basically immune to every type of magic, and have an AC of 20. They don't represent any pressing danger, so the wisest move is simply to ignore them.
SOCIETY: Will-o'-Wisps are evil beings that feed on the fear of their victims. They try to avoid combat and prefer to lure travelers into hazards or the lairs of more dangerous enemies.

Battle Devourer

RACE: Aberr. | ALIGNMENT: CE

CHALLENGE RATING: 11 | HIT POINTS: 78 | ARMOR CLASS: 30

Bred for use as spies and messengers in the Underdark, these former intellect Devourers are virtually unkillable, though they no longer retain their psionic abilities.

STR	15/2	INT	10/0	FOR	7
DEX	19/4	WIS	12/1	REF	7
CON	18/4	CHA	10/0	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** 6 x Gaze, Fear **Spells:** N/A

EQUIPMENT

Claw 1: Battle Devourer Claw (1d8 Dmg.; On Monster Hit: Slow) **Claw 2:** Battle Devourer Claw (1d8 Dmg.; On Monster Hit: Slow) **Special Attack:** N/A **Skin/Hide:** Battle Devourer Prop. (DR: +3 Soak 20 Dmg. Resist: Electrical Resist. 15/-; Immunity: Dmg. Type: Acid, Cold, Electrical, Fire, Sonic 100% Imm. Bonus) **Std. Eqp.:** N/A

Ettercap

RACE: Aberr. | ALIGNMENT: CN

CHALLENGE RATING: 4 | HIT POINTS: 27 | ARMOR CLASS: 13

Although not very intelligent, Ettercaps are cunning predators. Like the monstrous spiders that they keep as pets, they are skilled hunters and trappers.

STR	10/0	INT	6/-2	FOR	2
DEX	17/3	WIS	15/2	REF	4
CON	13/1	CHA	8/-1	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats: Other Feats:** Darkvision **Monster Abilities:** 3 x Bolt; Web **Spells:** Caster Level 5

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** Ettercap Bite (1d8 Dmg.; On Monster Hit: Poison Ettercap Venom) **Skin/Hide:** N/A **Std. Eqp.:** N/A

Gargoyle

RACE: M. Beast | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 38 | ARMOR CLASS: 16

Gargoyles are vicious predators that enjoy torturing weaker creatures. Their abiity to perch indefinitely without moving causes adventurers to frequently mistake them for statues.

STR	11/0	INT	6/-2	FOR	8
DEX	14/2	WIS	11/0	REF	6
CON	18/4	CHA	7/-2	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** Bite1d6 **Skin/Hide:** Gargoyle Prop. (DR: +1 Soak 15 Dmg.) **Std. Eqp.:** N/A

Gray Render

RACE: Beast | ALIGNMENT: N

CHALLENGE RATING: 8 | HIT POINTS: 125 | ARMOR CLASS: 19

At the same time both savage and maternal, the Gray Render is a deadly predator that often protects weaker creatures within its habitat.

STR	23/6	INT	3/-4	FOR	14
DEX	10/0	WIS	12/1	REF	7
CON	24/7	CHA	8/-1	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite 2d6 **Damage Skin/Hide:** N/A **Std. Eqp.:** N/A

Hook Horror

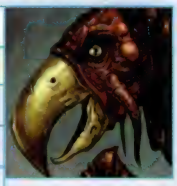
RACE: Aberr. ALIGNMENT: CN

CHALLENGE RATING: 4 HIT POINTS: 37 ARMOR CLASS: 17

Covered in a hard, chitinous exoskeleton, their forelimbs hooked and bladed, these vicious subterranean predators can be fearsome to behold.

STR	19/4	INT	6/-2	FOR	4
DEX	15/2	WIS	12/1	REF	3
CON	17/3	CHA	6/-2	WIL	5

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature); Weapon Focus (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d12 **Skin/Hide:** N/A **Std. Eqp.:** N/A

Intellect Devourer

RACE: Aberr. ALIGNMENT: CE

CHALLENGE RATING: 6 HIT POINTS: 39 ARMOR CLASS: 17

Bred by Mind-Flayers in the deepest Underdark, these "four-legged brains" stalk intelligent prey, co-opting their bodies after devouring their brains.

STR	13/1	INT	12/1	FOR	4
DEX	19/4	WIS	12/1	REF	6
CON	15/2	CHA	10/0	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature); Skill Focus (Perform) **Combat Feats:** N/A**Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 16

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Intellect Devourer Prop. (DR: +3 Soak 20 Dmg.; Dmg. Resist.: Electrical Resist. 15/-; Immunity: Dmg. Type: Fire 100% Imm. Bonus) **Std. Eqp.:** N/A

Krenshar

RACE: M. Beast ALIGNMENT: N

CHALLENGE RATING: 2 HIT POINTS: 11 ARMOR CLASS: 12

Though it hunts in packs, the fearsome Krenshar behaves much like other cats. its skeletal face, however, can be extremely frightening to the unwary.

STR	11/0	INT	6/-2	FOR	3
DEX	14/2	WIS	12/1	REF	5
CON	11/0	CHA	13/1	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 3 x Krenshar, Fear Gaze **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d2 **Claw 2:** Claw1d2 **Special Attack:** Bite1d2 **Skin/Hide:** N/A **Std. Eqp.:** N/A

Stirge

RACE: Beast ALIGNMENT: N

CHALLENGE RATING: 1/3 HIT POINTS: 5 ARMOR CLASS: 10

With the leathery, bat-like wings and the thirsty persistence of mosquitoes, Stirges rarely attack alone. In large swarms, they can be quite dangerous.

STR	3/-4	INT	3/-4	FOR	2
DEX	7/-2	WIS	12/1	REF	0
CON	10/0	CHA	6/-2	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Stirge Bite (1d4 Dmg.; On Monster Hit: Ability Drain Constitution) **Skin/Hide:** N/A **Std. Eqp.:** N/A

Umber Hulk

RACE: Aberr. ALIGNMENT: CE

CHALLENGE RATING: 7 HIT POINTS: 68 ARMOR CLASS: 17

Subterranean monsters that look like a cross between a giant ape and a beetle, Umber Hulks leave behind a wake of destuction wherever they go.

STR	23/6	INT	9/-1	FOR	6
DEX	13/1	WIS	11/0	REF	3
CON	19/4	CHA	13/1	WIL	6

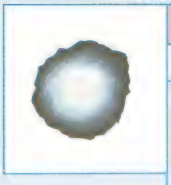
SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 6 x Gaze, Confusion **Spells:** N/A

EQUIPMENT

Claw 1: Claw2d4 **Claw 2:** Claw2d4 **Special Attack:** Bite2d8 **Skin/Hide:** N/A **Std. Eqp.:** N/A



Will-O'-Wisp

RACE: Aberr. ALIGNMENT: CE

CHALLENGE RATING: 8 HIT POINTS: 40 ARMOR CLASS: 20

Evil creatures that feed on the powerful emotions associated with panic, horror, and death, Will-O'-Wisps delight in luring travelers into deadly peril.

STR	3/-4	INT	15/2	FOR	3
DEX	29/9	WIS	16/3	REF	12
CON	10/0	CHA	12/1	WIL	9

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 1

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Will-O'-Wisp Shock **Skin/Hide:** Will-O'-Wisp Prop. (Immunity: Dmg. Type: Acid, Cold, Electrical, Fire, Sonic 100% Imm. Bonus; Immunity: Misc.: Death Magic, Mind-Affecting Spells, Paralysis) **Std. Eqp.:** N/A

PLANAR

GROUP: Celestials

STRATEGY: You'd have to be very bad indeed to run afoul of these powerful beings. If you do, skip the spells against the Hound Archons and Celestial Avengers, since their Spell Resistance will be a serious problem. Their Damage Reduction is only 10/+1, so any enchanted weapon will get past that and strike flesh. Their spells are generally not significant, with the Lightning Bolts of the Lantern Archons being a major exception. With only a handful of hit points, you should always hunt down the cowardly Lantern Archons first.

Damage Resistance spells like Stoneskin are worthless against Celestial Avengers, who wield +4 swords. Mages will be in trouble here.

SOCIETY: Celestials come from the planes of good, and they are examplars of goodness. They avoid harming good creatures when possible, but are certainly not pacifists. Celestials are vain and easily angered, and often act as self-appointed avengers of injustices. They are the arch-enemies of fiends.

Celestial Avenger

RACE: Outsider | ALIGNMENT: LG

CHALLENGE RATING: 11 | HIT POINTS: 66 | ARMOR CLASS: 27

These bear-headed heavenly creatures are fiercely independent, acting as self-appointed bounty hunters with the sole aim of bringing in the unrighteous for judgement.

STR	20/5	INT	16/3	FOR	9
DEX	17/3	WIS	16/3	REF	11
CON	13/1	CHA	16/3	WIL	11

SAVES

FEATS & ABILITIES

Active Combat Feats: Power attack **Class/Racial Feats:** Skill Focus (Concentration, Listen, Spot) **Combat Feats:** Cleave; Proficiency (martial); Weapon Specialization (longsword) **Other Feats:** Weapon Proficiency **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Celestial Avenger Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Paralysis; Saving Throw Bonus: Poison +4; Spell Resistance Bonus of 28 SR) **Std. Eqp.:** Celestial Avenger Sword

Half-Celestial Warrior

RACE: Outsider | ALIGNMENT: NG

CHALLENGE RATING: 14 | HIT POINTS: 99 | ARMOR CLASS: 13

STR	21/5	INT	12/1	FOR	10
DEX	15/2	WIS	16/3	REF	5
CON	16/3	CHA	14/2	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Alertness **Combat Feats:** Cleave; Point Blank Shot; Weapon Proficiency (exotic, martial, simple); Weapon Focus (longbow, longsword); Weapon Specialization (longsword) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Dodge; Great Fortitude; Shield Proficiency; Toughness **Other Feats:** Quick To Master **Monster Abilities:** N/A **Spells:** Caster Level 10

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Half-Celestial Prop. (Ability Bonus: Charisma +4, Const. +4, Dex. +2, Int. +2, Strength +4, Wisdom +4; Immunity: Dmg. Type: Acid Cold 100% Imm. Bonus, Electrical Bad Strref; Immunity: Misc.: Disease; Saving Throw Bonus: Poison +4; Spell Resistance Bonus of 28 SR) **Std. Eqp.:** Longsword +2; Half-Celestial Full Plate

Hound Archon (M/F)

RACE: Outsider | ALIGNMENT: LG

CHALLENGE RATING: 6 | HIT POINTS: 39 | ARMOR CLASS: 19

These hound-headed angels are the resolute footsoldiers of the celestial army. They are often sent to defend the innocent and protect the weak.

STR	15/2	INT	10/0	FOR	6
DEX	10/0	WIS	13/1	REF	5
CON	13/1	CHA	12/1	WIL	6

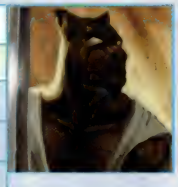
SAVES

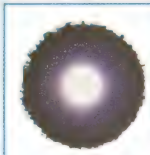
FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial) **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Menace **Spells:** Caster Level 6

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d8 **Skin/Hide:** Hound Archon Prop. (DR: +1 Soak 10 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Paralysis; Saving Throw Bonus: Poison +4; Spell Resistance Bonus of 16 SR) **Std. Eqp.:** Greatsword





Lantern Archon	Race: Outsider	Alignment: LG
Challenge Rating: 5	Hit Points: 4	Armor Class: 15

Often mistaken for more benevolent Will-o-Wisps, Lantern Archons are actually celestial beings. They are very friendly and usually eager to give what assistance they can.

STR	3/-4	INT	6/-2	Saves	
DEX	11/0	WIS	11/0		
CON	10/0	CHA	10/0		

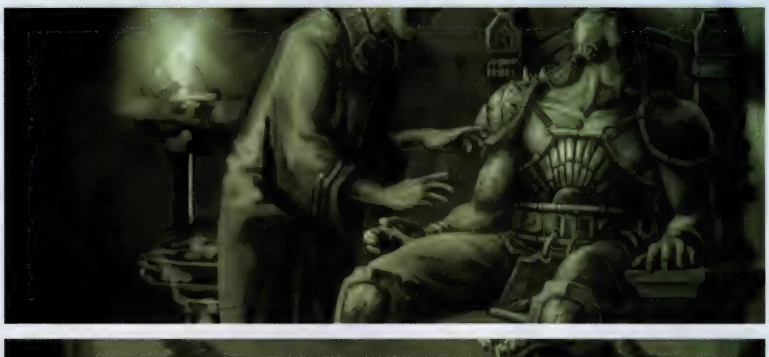
Saves	
FOR	2
REF	2
WIL	2

Feats & Abilities

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 6 x Bolt, Lightning **Spells:** Caster Level 3

Equipment

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Lantern Archon Prop. (DR: +1 Soak 20 Dmg.; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Paralysis; Saving Throw Bonus: Poison +4) **Std. Eqp.:** N/A



GROUP
Fiendish

BALOR

STRATEGY: Balors rock... They're powerful spellcasters, deadly warriors, and titans on defense. Oh, and they can summon more fiends to join in the fun, maybe even–I don't know–another Balor! Wouldn't that be fun?

Fighting a Balor is like fighting a Dragon. The Spell Resistance of 28 makes offensive spells completely worthless, so spellcasters will need to devote their spell slots to beefing up the fighters who will be stuck tasting Balor Sword. Everyone that can get a hit through the Damage Reduction should join in the fight, because it's imperitive you prevent the Balor's offensive spells and summoning abilities.

SOCIETY: King of the Tanar'ri family of Abyss-dwelling demons, Balors live to spread fear and misery.

HALF-FIEND WARRIOR

STRATEGY: Just like the Half-Dragons, but half demon. The default Half-Fiend is a Barbarian, but there's no reason you couldn't make a Half-Fiend of any class.

DOGS (HELL HOUND/HOUND OF XVIM/SHADOW MASTIFF)

STRATEGY: As long as you stay away from the fire immunities of the hounds, these creatures are fairly easy to defeat. A bit of spell preparation can protect you from Fire Breath and Fear effects, but parties without magic can always simply overwhelm these deadly canids; they have powerful attacks but aren't particularly tough on defense.

SOCIETY: Hell Hounds are summoned from other dimensions to be servants to powerful evil beings. Shadow Mastiffs come from the plane of shadow and hate the light. They are usually independent hunters, not under the control of any other beings. Xvim is an evil deity that enjoys causing trouble by infusing a bit of himself into wild beasts, twisting them into an eviler and more powerful version of the animal in question.

RAKSHASA

STRATEGY: Immune to spells of Level 8 or lower? Damage Reduction 20/+3? That's better than most dragons, but Rakshasa don't have the goods on offense. Their spells are powerful but not game enders, and their natural attacks and sword strikes do little damage. If you can get past the Damage Reduction, keep this fight in melee range for a fairly easy win.

SOCIETY: Rakshasa are evil, intelligent, selfish beings that value luxury and are more than willing to acquire it at the expense of others.

SUCCUBUS

STRATEGY: Succubi are basically spellcasters that have little recourse when surrounded by a party of fighters and attacked repeatedly with +2 weapons. Use your spells to enhance your weapons and defend against her spells, not to attack her directly. With her various resistances, immunities and Spell Resistances, offensive spells simply are not effective.

SOCIETY: Succubi are a type of Tanar'ri demons from the foul plane of the Abyss. They are chaotic evil beings that live to tempt men to evil, and shy away from physical combat.

VROCK

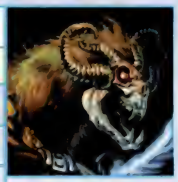
STRATEGY: These nasty demons have a number of deadly abilities. Vrock Spores are a damage-dealing poison that affect everyone within range of the demon, and are one of a few reasons that a group rush strategy is not the best idea here. The other is Mass Charm, which can wipe out a party that isn't magically defended. Better to divide up your party and not have more than two or three fighters going after one Vrock at a time. As with Balors, Vrock have a high SR so use your spells defensively. Magic Aura Against Alignment is, as always, a godsend.

SOCIETY: Another Tanar'ri species, Vrock serve as guards to stronger demons and sorcerers powerful enough to summon them.

BALOR	RACE: Outsider	ALIGNMENT: CE
CHALLENGE RATING: 17	HIT POINTS: 110	ARMOR CLASS: 30

With their flaring wings and skull-like countenance, these demons are among the greatest and most feared of their kind.

STR	25/7	INT	20/5	SAVES FOR	12
DEX	13/1	WIS	20/5	REF	9
CON	19/4	CHA	16/3	WIL	13



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial) **Other Feats:** Darkvision **Monster Abilities:** 1 x Summon Tanarri **Spells:** Caster Level 20

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Balor Slam (1d6 Dmg.; On Monster Hit: Fear) **Skin/Hide:** Balor Prop. (DR: +3 Soak 30 Dmg.; Dmg. Resist.: Acid, Cold, Fire Resist 20/-; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Poison Spell Resistance of 28 SR) **Std. Eqp.:** Balor Sword

Balor Lord

RACE: Outsider | ALIGNMENT: CE

CHALLENGE RATING: 18 | HIT POINTS: 110 | ARMOR CLASS: 30

With their flaring wings and skull-like countenance, these demons are among the greatest and most feared of their kind..

STR	25/7	INT	20/5	FOR	12
DEX	13/1	WIS	20/5	REF	9
CON	19/4	CHA	16/3	WIL	13

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (cretaure) **Combat Feats:** Cleave; Weapon Proficiency (martial, exotic) **Other Feats:** Darkvision **Monster Abilities:** 4 x Summon Tanarri **Spells:** Caster Level 20

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Balor Slam (1d6 Dmg.; On Monster Hit: Fear) **Skin/Hide:** Balor Prop. (DR: +3 Soak 30 Dmg.; Dmg. Resist.: Acid, Cold, Fire Resist 20/-; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Poison Spell Resistance of 28 SR) **Std. Eqp.:** Balor Lord Sword

Half-Fiend Warrior

RACE: Outsider | ALIGNMENT: CE

CHALLENGE RATING: 12 | HIT POINTS: 100 | ARMOR CLASS: 15

STR	26/8	INT	10/0	FOR	10
DEX	18/4	WIS	12/1	REF	7
CON	17/3	CHA	8/-1	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** N/A **Combat Feats:** Cleave; Weapon Proficiency (martial, simple); Weapon Focus (greataxe) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Shield Proficiency **Other Feats:** Barbarian Fast Movement; Darkvision; Uncanny Dodge 1, 2, 3 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: Half-Dragon/Half-Fiend Bite **Claw 2:** Half-Dragon/Half-Fiend Claw **Special Attack:** N/A **Skin/Hide:** Half-Fiend Prop. (Ability Bonus; Char., Con.: +2, Dex., Int., Str.: +4; Dmg. Resist.: Fire Resist 20/-; Immunity: Dmg. Type: Acid, Cold 100 % Imm. Bonus, Electrical Bad Sttref; Immunity: Misc.: Poison) **Std. Eqp.:** Greataxe +2; Half-Fiend Studded Leather Armor

Hell Hound

RACE: Outsider | ALIGNMENT: CG

CHALLENGE RATING: 3 | HIT POINTS: 22 | ARMOR CLASS: 16

Hell Hounds are aggressive, fire-breathing canines brought to the Prime Material plane from the depths of the Abyss to serve their evil masters.

STR	13/1	INT	6/-2	FOR	5
DEX	13/1	WIS	10/0	REF	5
CON	13/1	CHA	6/-2	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 3 x Hell Hound Fire Breath **Spells:** N/A

EQUIPMENT

Claw 1: Bite1d8 **Claw 2:** N/A **Special Attack:** Slam3d8 **Skin/Hide:** Fire Giant / Hell Hound Prop. (DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Immunity: Misc.: N/A) **Std. Eqp.:** N/A

Hound of Xvim

RACE: Outsider | ALIGNMENT: LE

CHALLENGE RATING: 5 | HIT POINTS: 27 | ARMOR CLASS: 16

Touched by the evil deity, Iyatchu Xvim, this hound has undergone a horrible transformation to become an agent of his infernal will.

STR	19/4	INT	6/-2	FOR	5
DEX	13/1	WIS	10/0	REF	5
CON	13/1	CHA	14/2	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 4 x Hell Hound Fire Breath; 1 x Howl, Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite2d6 **Skin/Hide:** Hound of Xvim Prop. (Immunity: Misc.: Fear, Poison) **Std. Eqp.:** N/A

Rakshasa (M/F)

RACE: Outsider | ALIGNMENT: LE

CHALLENGE RATING: 8 | HIT POINTS: 52 | ARMOR CLASS: 21

Humanoid tigers garbed in the clothes of nobility, Rakshasa use their considerable powers to maintain their decadent lifestyle at others' expense.

				SAVES		
STR	12/1	INT	13/1		FOR	8
DEX	14/2	WIS	13/1		REF	7
CON	16/3	CHA	17/3		WIL	6

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (exotic, martial) **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 7

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d6 **Skin/Hide:** Rakshasa Prop. (DR: +3 Soak 20 Dmg.; Immunity: Spells by Level: Level 8 or Lower) **Std. Eqp.:** Rapier +1 Female; Bastard Sword +1 Male

Shadow Mastiff

RACE: Outsider | ALIGNMENT: CN

CHALLENGE RATING: 4 | HIT POINTS: 30 | ARMOR CLASS: 13

Large, black hounds that prowl the night, these extra-planar creatures seek out prey they can find. While not intelligent, they are cunning.

				SAVES		
STR	17/3	INT	4/-3		FOR	7
DEX	13/1	WIS	12/1		REF	5
CON	17/3	CHA	13/1		WIL	5

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** 3 x Howl, Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Std. Eqp.:** N/A

Succubus

RACE: Outsider | ALIGNMENT: CE

CHALLENGE RATING: 7 | HIT POINTS: 33 | ARMOR CLASS: 20

With perfect builds and flawless skin, Succubi are the most comely of demons. They live to tempt mortal men into crimes of passion.

				SAVES		
STR	13/1	INT	16/3		FOR	6
DEX	13/1	WIS	14/2		REF	6
CON	13/1	CHA	20/5		WIL	7

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** 1 x Summon Tanarri **Spells:** _____

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Succubus Prop. (DR: +2 Soak 20 Dmg.; Dmg. Resist.: Acid, Cold, Fire Resist. 20/-; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity:Misc.: Poison; Spell Resistance Bonus of 12 SR) **Std. Eqp.:** N/A

Vrock

RACE: Outsider | ALIGNMENT: CE

CHALLENGE RATING: 11 | HIT POINTS: 60 | ARMOR CLASS: 25

Bony, vulture-like creatures, Vrocks are the foot soldiers of the Abyss. Notoriously vain, they prance about in battle, admiring their sharp claws and beaks.

				SAVES		
STR	19/4	INT	14/2		FOR	9
DEX	15/2	WIS	14/2		REF	8
CON	17/3	CHA	12/1		WIL	8

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 3 x Pulse, Vrock Spores; 1 Summon Tanarri **Spells:** Caster Level 12

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite1d6 **Skin/Hide:** Vrock Prop. (DR: +2 Soak 20 Dmg.; Dmg. Resist.: Acid, Cold, Fire Resist. 20/-; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Poison; Spell Resistance Bonus of 22 SR) **Std. Eqp.:** N/A

STRATEGY: Imps and Quasits are very similar beings. Both like to cast Invisibility early and are rarely seen until they attack. Both can poison their victims with a venom that reduces their Dexterity scores. Both have moderate Spell Resistances, good AC's, inconsequential Regeneration abilities, and are resistant to Fire. Both are annoying, but neither are a serious threat.

Mephits are no tougher, although the damage from their elemental attacks can be painful if you're forced to fight them in large groups. All Mephits have Damage Reduction 5/+1, and Regeneration +2, which can be a solid defense against characters without +1 weapons. But they can all be easily defeated with spells, provided you don't use their own type of element against them.

SOCIETY: Imps and Quasits are evil creatures from the Abyss, the same plane as the Tanar'ri demons. Mephits are from the various elemental planes, and are usually neutral, not inherently evil.

GROUP
Imps

Air Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 17

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	10/0	INT	12/1	FOR	3
DEX	17/3	WIS	11/0	REF	6
CON	10/0	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 1

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Mephit Prop. (DR: +1 Soak 5 Dmg.; Regeneration +2) **Std. Eqp.:** N/A

Dust Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 2 | HIT POINTS: 13 | ARMOR CLASS: 17

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	10/0	INT	12/1	FOR	3
DEX	17/3	WIS	11/0	REF	6
CON	10/0	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** _____

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Mephit Prop. (DR: +1 Soak 5 Dmg.; Regeneration +2) **Std. Eqp.:** N/A

Earth Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 16 | ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	17/3	INT	12/1	FOR	4
DEX	8/-1	WIS	11/0	REF	2
CON	13/1	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** _____

EQUIPMENT

Claw 1: Claw1d3**Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Earth/Salt Mephit Prop. (DR: +1 Soak 10 Dmg.) **Std. Eqp.:** N/A

Fire Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	10/0	INT	12/1	FOR	3
DEX	13/1	WIS	11/0	REF	4
CON	10/0	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Cone of Fire **Spells:** Caster Level 3

EQUIPMENT

Claw 1: Fire/Magma/Steam Mephit Claw (1d3 Dmg.; Dmg. Bonus: Fire 2 Dmg.) **Claw 2:** Fire/Magma/Steam Mephit Claw (1d3 Dmg.; Dmg. Bonus: Fire 2 Dmg.) **Special Attack:** N/A **Skin/Hide:** Fire/Magma/Steam Mephit Prop. (DR: +1 Soak 5 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Regeneration +2) **Std. Eqp.:** N/A

Ice Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 18

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	10/0	INT	12/1	FOR	3
DEX	17/3	WIS	11/0	REF	6
CON	10/0	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 1 / Caster Level 3

EQUIPMENT

Claw 1: Ice Mephit Claw (1d3 Dmg.; Dmg. Bonus: Cold 2 Dmg.) **Claw 2:** Ice Mephit Claw (1d3 Dmg.; Dmg. Bonus: Cold 2 Dmg.) **Special Attack:** N/A **Skin/Hide:** Ice Mephit Prop. (DR: +1 Soak 5 Damage; DV: Fire 50% Dmg. Vuln.; Immunity: Dmg. Type: Cold100% Imm. Bonus; Regeneration +2) **Std. Eqp.:** N/A

Imp

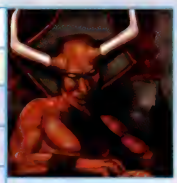
RACE: Outsider | ALIGNMENT: CG

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 18

Small but insidious devils often bound to the service of Sorcerers and Wizards, Imps have a gleeful attraction to pranks, riddles, and bawdy jokes.

STR	10/0	INT	10/0	FOR	3
DEX	17/3	WIS	12/1	REF	6
CON	10/0	CHA	10/0	WIL	4

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 5 / Caster Level 6

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Imp Stinger (1d4 Dmg.; On Monster Hit: Poison Quasit Venom) **Skin/Hide:** Imp Prop. (DR: +1 Soak 5 Dmg.; Dmg. Resist.: Fire Resist. 20/-; Immunity: Misc.: Poison; Regeneration +2; Spell Resistance of 10 SR) **Std. Eqp.:** N/A

Magma Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	10/0	INT	12/1	FOR	3
DEX	13/1	WIS	11/0	REF	4
CON	10/0	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Cone of Fire **Spells:** _____

EQUIPMENT

Claw 1: Fire/Magma/Steam Mephit Claw (1d3 Dmg.; Dmg. Bonus: Fire 2 Dmg.) **Claw 2:** Fire/Magma/Steam Mephit Claw (1d3 Dmg.; Dmg. Bonus: Fire 2 Dmg.) **Special Attack:** N/A **Skin/Hide:** Fire/Magma/Steam Mephit Prop. (DR: +1 Soak 5 Damage; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Regeneration +2) **Std. Eqp.:** N/A

Ooze Mephit

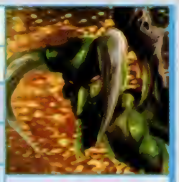
RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 16 | ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	14/2	INT	12/1	FOR	4
DEX	10/0	WIS	11/0	REF	3
CON	13/1	CHA	15/2	WIL	3

SAVES



FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Cone of Acid **Spells:** Caster Level 3 / Caster Level 6

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Mephit Prop. (DR: +1 Soak 5 Dmg.; Regeneration +2) **Std. Eqp.:** N/A

Quasit

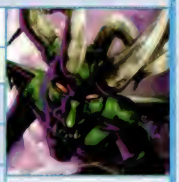
RACE: Outsider | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 17

Cowardly but cunning, these small, demonic humanoids are often bound to the will of sorcerers and wizards and made to do their bidding.

STR	8/-1	INT	10/0	FOR	3
DEX	17/3	WIS	12/1	REF	6
CON	10/0	CHA	10/0	WIL	4

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Defensive Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 5 / Caster Level 6

EQUIPMENT

Claw 1: Quasit Claw (1d3 Dmg.; On Monster Hit: Poison Quasit Venom) **Claw 2:** Quasit Claw (1d3 Dmg.; On Monster Hit: Poison Quasit Venom) **Special Attack:** Bite1d4 **Skin/Hide:** Quasit Prop. (Dmg. Resist.: Fire Resist. 20/-; Immunity: Misc.: Poison; Regeneration +2; Spell Resistance of 10 SR) **Std. Eqp.:** N/A

Salt Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 16 | ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	17/3	INT	12/1	FOR	4
DEX	8/-1	WIS	11/0	REF	2
CON	13/1	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Mephit, Salt Breath **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** N/A **Skin/Hide:** Earth/Salt Mephit Prop. (DR: +1 Soak 10 Dmg.) **Std. Eqp.:** N/A

Steam Mephit

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 13 | ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

STR	10/0	INT	12/1	FOR	3
DEX	13/1	WIS	11/0	REF	4
CON	10/0	CHA	15/2	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Mephit, Steam Breath **Spells:** N/A

EQUIPMENT

Claw 1: Fire/Magma/Steam Mephit Claw (1d3 Dmg.; Dmg. Bonus: Fire 2 Dmg.) **Claw 2:** Fire/Magma/Steam Mephit Claw (1d3 Dmg.; Dmg. Bonus: Fire 2 Dmg.) **Special Attack:** N/A **Skin/Hide:** Fire/Magma/Steam Mephit Prop. (DR: +1 Soak 5 Dmg.; DV: Cold 50% Dmg. Vuln.; Immunity: Dmg. Type: Fire 100% Imm. Bonus; Regeneration +2) **Std. Eqp.:** N/A

WATER MEPHIT	RACE: Outsider	ALIGNMENT: NE
CHALLENGE RATING: 3	HIT POINTS: 16	ARMOR CLASS: 16

Chaotic and destructive, Mephits come in many shapes and have a wide range of abilities. Most are drawn from the various Elemental and Quasi-Elemental Planes.

				SAVES	
STR	14/2	INT	12/1	FOR	4
DEX	10/0	WIS	11/0	REF	3
CON	13/1	CHA	15/2	WIL	3

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Cone of Acid **Spells:** ——

EQUIPMENT

Claw 1: 1d3 **Claw 2:** 1d3 **Special Attack:** N/A **Skin/Hide:** Mephit Prop. (DR: +1 Soak 5 Dmg.; Regeneration +2) **Std. Eqp.:** N/A

GROUP
Others

BLINK DOG

STRATEGY: Blink Dogs have the same teleport-to-weakest-member script as Phase Spiders do. So keep your HP up or you could end up facing a whole pack of these crafty, dangerous beasts. They have no special properties outside of their teleportation habit.

SOCIETY: Blink Dogs are good doggies. Yes they are. Good little doggies. Lawful Good little doggies. They travel in packs of 7-16 members. They are only hostile to evil beings that threaten their territories.

FENHOUND

STRATEGY: The inter-dimensional nature of Fenhounds makes them pretty hard to hit. SR 16 and Damage Resistance 10/+2 are not attributes you often see on a creature with a Challenge Rating of 4. Their only attack is a 1d10+4 bite, so they aren't particularly dangerous, but spellcasters will be flustered by their Spell Resistance and fighters will want to use +2 or better weapons to get around the Damage Reduction. A better strategy might be to avoid pissing them off in the first place.

SOCIETY: Fenhounds are also good doggies. They should not attack unless you threaten the place they were called upon to guard or committed some wrong they have been asked to avenge.

TIEFLING

STRATEGY: Tieflings are humans who have small amounts of fiend blood in their ancestry. This blood carries with it an evil alignment (usually), a limited amount of elemental Damage Resistance, +2 to Spot and Listen checks, and the ability to cast Darkness once per day. The sample Tieflings in the game are very weak, but you can feel free to give their properties to any Tiefling NPC.

Oddly missing in Neverwinter Nights are the Aasimar, humans with Celestial Ancestry. You could make an Aasimar the same way you made a Tiefling, but with a good alignment and have the ability to cast Light instead.

Blink Dog

RACE: M. Beast | ALIGNMENT: LG

CHALLENGE RATING: 3 | HIT POINTS: 22 | ARMOR CLASS: 16

An intelligent, predatory canine with limited teleportation abilities, Blink Dogs hunt in packs, 'blinking' randomly until they surround their prey.

STR	10/0	INT	10/0	FOR	4
DEX	17/3	WIS	13/1	REF	7
CON	10/0	CHA	11/0	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature); Weapon Specialization (creature) **Combat Feats:** N/A **Defensive Feats:** Iron Will **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Bite1d6 **Skin/Hide:** N/A **Std. Eqp.:** N/A

Fenhound

RACE: Outsider | ALIGNMENT: CG

CHALLENGE RATING: 4 | HIT POINTS: 32 | ARMOR CLASS: 17

Fenhounds serve as avenging spirits and guardians of the darkest moors and bogs. They draw their power from the mists of the places they protect.

STR	19/4	INT	3/-4	FOR	6
DEX	17/3	WIS	12/1	REF	7
CON	15/2	CHA	10/0	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Howl, Fear **Spells:** N/A

EQUIPMENT

Claw 1: Bite1d10 **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Fenhound Prop. (DR: +2 Soak 10 Dmg.; Spell Resistance Bonus of 16 SR) **Std. Eqp.:** N/A

Tiefling (M/F)

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 1/2 | HIT POINTS: 4 | ARMOR CLASS: 11

Facing widely held stereotypes of being twisted, devious, or untrustworthy, all Tieflings can trace their bloodline back to some form of fiendish origin.

STR	10/0	INT	13/1	FOR	2
DEX	13/1	WIS	11/0	REF	3
CON	10/0	CHA	8/-1	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency (martial); Weapon Finesse **Defensive Feats:** Armor Proficiency (light) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 1

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Tiefling Prop. (Dmg. Resist.: Cold, Electrical, Fire Resist. 5/-) **Std. Eqp.:** Rapier; Studded Leather Mail; Small Shield

GROUP: Slaadi

STRATEGY: Slaadi are fearsome creatures that spell the violent end of a module for many players. When you encounter even a single Slaad, you'll need to hit him with everything you can, because any Slaad (except red) can summon in another Slaad. This can result in things quickly getting out of hand. Compounding this problem is the fact that Slaadi don't die easily. All Slaadi have Damage Resistance that will knock the first 5 damage off any elemental magic attack and Regeneration that will heal another 5 points a turn. The more powerful ones have Damage Reduction as well, although by the time you run into Slaad you should be high enough level to have weapons that will get around that. Death Slaad, Gray Slaad, and Green Slaad are all spellcasters with nasty spell lists. Focus your attacks on them to fizzle their spells; as creatures of chaos, their Concentration scores are crap.

Red Slaad and Blue Slaad physical attacks may result in a nasty disease like Red Slaad Eggs or Cackle Fever, so make sure you have some solution ready or you could survive the battle only to be too weakened to survive encounters on the way home. Remember that diseases take time to gestate, so save the treatments for the end of the battle.

SOCIETY: Slaadi are natives of Limbo, the plane of chaos, where they roam in packs of 2 to 10 members, hunting prey and battling with each other. In mating seasons, they all converge upon a place called the Spawning Stone, where they do disgusting things that result in the birth of more Slaadi. Since the chaotic nature of limbo would be impossible to create in Neverwinter Nights, any Slaadi encountered by a player are likely to have been summoned by chaotic evil beings, or rogue packs fleeing Limbo to escape from a rival tribe.

Red Slaadi do not usually associate with each other. Blue Slaadi often fight as a group, and spell-casting green and grey Slaadi will work together when necessary. All Slaadi fear and obey the Death Slaadi.

Blue Slaad

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 7 | HIT POINTS: 60 | ARMOR CLASS: 18

Creatures of chaos, Blue Slaadi are bullies that value only strength and power. They are fearsome in battle and are more agile than they appear.

STR	19/4	INT	6/-2	FOR	9
DEX	15/2	WIS	6/-2	REF	8
CON	17/3	CHA	10/0	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** 1 x Summon Slaad **Spells:** Caster Level 10

EQUIPMENT

Claw 1: Claw2d6 **Claw 2:** Claw2d6 **Special Attack:** Blue Slaad Bite (2d8 Dmg.; On Monster Hit: Disease Cackle Fever) **Skin/Hide:** Red/Blue/Green Slaad Prop. (Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

Death Slaad

RACE: Outsider | ALIGNMENT: CN

CHALLENGE RATING: 15 | HIT POINTS: 112 | ARMOR CLASS: 25

Creatures of chaos, Death Slaadi are the rulers of their frog-like kind. Aside from their considerable magical powers, they are also gifted fighters.

STR	20/5	INT	18/4	FOR	12
DEX	18/4	WIS	18/4	REF	13
CON	17/3	CHA	18/4	WIL	13

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Summon Slaad **Spells:** Caster Level 18

EQUIPMENT

Claw 1: Claw3d6 **Claw 2:** Claw3d6 **Special Attack:** Bite2d10 **Skin/Hide:** Death Slaad Prop. (DR: +2 Soak 20 Dmg.; Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

Death Slaad Lord

RACE: Outsider | ALIGNMENT: CN

CHALLENGE RATING: 15 | HIT POINTS: 112 | ARMOR CLASS: 25

Creatures of chaos, Death Slaadi are the rulers of their frog-like kind. Aside from their considerable magical powers, they are also gifted fighters.

STR	20/5	INT	18/4	FOR	12
DEX	18/4	WIS	18/4	REF	13
CON	17/3	CHA	18/4	WIL	13

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 3 X Aura of Menace; 4 x Summon Slaad **Spells:** Caster Level18

EQUIPMENT

Claw 1: Claw3d6 **Claw 2:** Claw3d6 **Special Attack:** Bite2d10 **Skin/Hide:** Death Slaad Prop. (DR: +2 Soak 20 Dmg.; Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

Gray Slaad

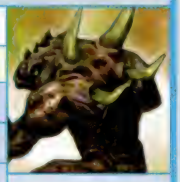
RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 10 | HIT POINTS: 75 | ARMOR CLASS: 21

Creatures of chaos, Gray Slaadi are the wizards of their frog-like kind. They enjoy crafting items of dire magic to further their own power.

STR	19/4	INT	14/2	FOR	10
DEX	13/1	WIS	14/2	REF	8
CON	17/3	CHA	14/2	WIL	9

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Summon Slaad **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d4 **Claw 2:** Claw2d4 **Special Attack:** Bite 2d8 **Skin/Hide:** Gray Slaad Prop. (DR: +1 Soak 10 Dmg.; Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

Gray Slaad Lord

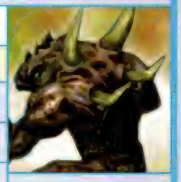
RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 10 | HIT POINTS: 75 | ARMOR CLASS: 21

Creatures of chaos, Gray Slaadi are the wizards of their frog-like kind. They enjoy crafting items of dire magic to further their own power.

STR	19/4	INT	14/2	FOR	10
DEX	13/1	WIS	14/2	REF	8
CON	17/3	CHA	14/2	WIL	9

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 3 x Aura of Blinding; 4 x Summon Slaad **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Claw2d4 **Claw 2:** Claw2d4 **Special Attack:** Bite2d8 **Skin/Hide:** Gray Slaad Prop. (DR: +1 Soak 10 Dmg.; Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

Green Slaad

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 8 | HIT POINTS: 67 | ARMOR CLASS: 20

Creatures of chaos, Green Slaadi are self-centered, arrogant louts. They lust after magical power and are often in pursuit of great artifacts.

STR	19/4	INT	10/0	FOR	9
DEX	13/1	WIS	10/0	REF	7
CON	17/3	CHA	10/0	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** 1 x Summon Slaad **Spells:** Caster Level 12

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite2d8 **Skin/Hide:** Red/Blue/Green Slaad Prop. (Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

Red Slaad

RACE: Outsider | ALIGNMENT: NE

CHALLENGE RATING: 6 | HIT POINTS: 52 | ARMOR CLASS: 16

Creatures of chaos, Red Slaadi are the weakest of their kind. Most seek escape from the cruel treatment offered by their more powerful kin.

STR	19/4	INT	6/-2	FOR	8
DEX	13/1	WIS	6/-2	REF	6
CON	17/3	CHA	8/-1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** 1 x Howl, Stunning; 1 x Summon Slaad **Spells:** N/A

EQUIPMENT

Claw 1: Red Slaad Claw (1d4 Dmg.; On Monster Hit: Disease Red Slaad Eggs) **Claw 2:** Red Slaad Claw (1d4 Dmg.; On Monster Hit: Disease Red Slaad Eggs) **Special Attack:** Bite2d8 **Skin/Hide:** Red/Blue/Green Slaad Prop. (Dmg. Resist.: Acid, Cold, Electrical, Fire, Sonic Resist. 5/-; Regeneration +5) **Std. Eqp.:** N/A

SHAPECHANGERS

ARANEA

STRATEGY: Aranea are shapechanging spiders that can take the form of both a humanoid and a human/spider hybrid. Currently, they lack any abilities that would allow them to do so, and have no special scripts, although this may change before the game ships. If not, gifted scripters could make a version of the lycanthrope script and bring the Aranea's special ability to life. Without that, it's just another spider with the added bonus of Damage Reduction 10/+1.

LYCANTHROPES

STRATEGY: All three Lycanthropes appear as normal Humans (the three Bandits), until attacked. Then a custom script polymorphs them into their second form (module builders could give this script to a different NPC to change their appearance when in human form). Their animal forms range in power, but all gain Damage Reduction 10/+1, which is a huge factor for low Level players. Use +1 weapons or spells or they'll wear you down with their physical attacks.

SOCIETY: Some Lycanthropes were birthed naturally, but most were the victims of a bite from someone else who had the affliction. Being bitten does not change ones abilities as a human, but may change their alignment.

Aranea

RACE: Shpchng. | ALIGNMENT: N

CHALLENGE RATING: 3 | HIT POINTS: 19 | ARMOR CLASS: 13

Intelligent, shapeshifting spiders with sorcerous powers, Aranea avoid physical combat, prefering to rely on their webs and spells instead.

STR	11/0	INT	14/2	FOR	5
DEX	15/2	WIS	13/1	REF	5
CON	14/2	CHA	14/2	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 2

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Aranea Bite (1d6 Dmg.; On Monster Hit: Poison Aranea Venom) **Skin/Hide:** Shapechanger Prop. (DR: +1 Soak 10 Dmg) **Std. Eqp.:** N/A

Bandit

RACE: Human | ALIGNMENT: CN

CHALLENGE RATING: 1/2 | HIT POINTS: 12 | ARMOR CLASS: 11

STR	15/2	INT	10/0	FOR	3
DEX	13/1	WIS	12/1	REF	1
CON	12/1	CHA	8/-1	WIL	1

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness- **Combat Feats:** Weapon Proficiency (exotic, martial, simple); Weapon Focus (bastard sword) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency; Toughness **Other Feats:** Quick To Master **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** N/A **Std. Eqp.:** N/A

Werecat

RACE: Shpchng. ALIGNMENT: NE

CHALLENGE RATING: 5 HIT POINTS: 45 ARMOR CLASS: 17

In their hybrid form, Werecats are selfish and manipulative. They have been known to toy with and enslave their prey, often driving them to madness.

				SAVES		
STR	23/6	INT	10/0		FOR	8
DEX	15/2	WIS	10/0		REF	7
CON	17/3	CHA	10/0		WIL	5

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature); Weapon Focus (creature); Weapon Specialization (creature) N/A **Combat Feats:** Cleave **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** Bite2d6 **Skin/Hide:** Shapechanger Prop. (DR: +1 Soak 10 Dmg.) **Std. Eqp.:** N/A

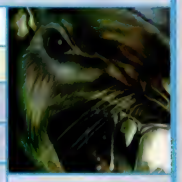
Wererat

RACE: Shpchng. ALIGNMENT: CG

CHALLENGE RATING: 1/2 HIT POINTS: 5 ARMOR CLASS: 16

In their hybrid form, Wererats are secretive and untrustworthy. They make excellent thieves and spies, however -- if you're willing to risk being double-crossed.

				SAVES		
STR	10/0	INT	10/0		FOR	3
DEX	17/3	WIS	10/0		REF	5
CON	13/1	CHA	10/0		WIL	2



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature); Weapon Focus (creature); Weapon Specialization (creature) N/A **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite1d4 **Skin/Hide:** Shapechanger Prop. (DR: +1 Soak 10 Dmg.) **Std. Eqp.:** N/A

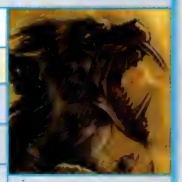
Werewolf

RACE: Shpchng. ALIGNMENT: CE

CHALLENGE RATING: 2 HIT POINTS: 13 ARMOR CLASS: 16

The most common of the lycanthropes, Werewolves are also the most complex, socially. While some lead solitary existences, others form complicated ritual-bound societies.

				SAVES		
STR	13/1	INT	10/0		FOR	5
DEX	15/1	WIS	10/0		REF	5
CON	15/2	CHA	10/0		WIL	3



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature); Weapon Focus (creature); Weapon Specialization (creature) N/A **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d3 **Claw 2:** Claw1d3 **Special Attack:** Bite1d6 **Skin/Hide:** Shapechanger Prop. (DR: +1 Soak 10 Dmg.) **Std. Eqp.:** N/A

UNDEAD

GROUP Ghouls

STRATEGY: Ghouls have Turn Resistance, making them unusually difficult for Clerics to deal with. Their physical hits can Stun their victims for a round, and Ghasts also have a poisonous gas attack. Ghoul Ravagers can be particularly savage, with the combination of casting Haste on themselves and using Stun attacks on you keeping a character on the ropes for a long time. All Ghouls have weak Fortitude and Reflex saving throws and are susceptible to damage-dealing magics.

SOCIETY: Ghouls primarily haunt battlefields and other places with a large concentration of corpses. They eat the flesh of the dead, which has lead to the belief that Ghouls are the spirits of dead cannibals.

GHAST

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 4 | HIT POINTS: 26 | ARMOR CLASS: 16

More deadly and cunning than their lesser kin, the Ghoul, these undead carrion eaters have the added ability to paralyze their victims.

				SAVES	
STR	13/1	INT	13/1	FOR	1
DEX	15/2	WIS	14/2	REF	3
CON	10/0	CHA	16/3	WIL	6

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** 1x Tyrant Fog Zombie Mist **Spells:** N/A

EQUIPMENT

Claw 1: Ghast Claw (1d4 Dmg.; On Monster Hit: Stun) **Claw 2:** Ghast Claw (1d4 Dmg.; On Monster Hit: Stun) **Special Attack:** Ghast Bite (1d8 Dmg.; On Monster Hit: Stun) **Skin/Hide:** Ghoul/Ghast Prop. (Immunity: Misc.: Critical Hits, Death Magic, Level/Ability Drain, Mind-Affecting Spells, Poison; Turn Resistance +2) **Std. Eqp.: N/A**

GHOUL

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 2 | HIT POINTS: 13 | ARMOR CLASS: 14

These undead carrion eaters haunt the graveyards and battlefields that bore them, still bearing the rotting flesh they once wore in life.

				SAVES	
STR	13/1	INT	13/1	FOR	0
DEX	15/2	WIS	14/2	REF	2
CON	10/0	CHA	16/3	WIL	5

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Ghoul Claw (1d3 Dmg.; On Monster Hit: Stun) **Claw 2:** Ghoul Claw (1d3 Dmg.; On Monster Hit: Stun) **Special Attack:** Ghoul Bite (1d6 Dmg.; On Monster Hit: Stun) **Skin/Hide:** Ghoul/Ghast Prop. (Immunity: Misc.: Critical Hits, Death Magic, Level/Ability Drain, Mind-Affecting Spells, Poison; Turn Resistance +2) **Std. Eqp.: N/A**

GHOUL LORD

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 5 | HIT POINTS: 39 | ARMOR CLASS: 14

Corrupt officials are often said to become Ghoul Lords in their unlife. Cunning and intelligent, they often maintain a retinue of undead servants.

				SAVES	
STR	13/1	INT	13/1	FOR	2
DEX	15/2	WIS	14/2	REF	4
CON	10/0	CHA	16/3	WIL	7

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Improved Critical (creature); Proficiency (creature) **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Menace **Spells:** N/A

EQUIPMENT

Claw 1: Ghoul Lord Claw (1d6 Dmg.; On Monster Hit: Stun) **Claw 2:** Ghoul Lord Claw (1d6 Dmg.; On Monster Hit: Stun) **Special Attack:** Ghoul Lord Bite (1d10 Dmg.; On Monster Hit: Stun) **Skin/Hide:** Ghoul Lord Prop. (Immunity: Misc.: Critical Hits, Death Magic, Level/Ability Drain, Mind-Affecting Spells, Poison; Turn Resistance +4) **Std. Eqp.: N/A**

Ghoul Ravager

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 8 | HIT POINTS: 130 | ARMOR CLASS: 12

These undead carrion eaters haunt the graveyards and battlefields that bore them, still bearing the rotting flesh they once wore in life.

STR	13/1	INT	13/1	FOR	3
DEX	15/2	WIS	14/2	REF	5
CON	10/0	CHA	16/3	WIL	8

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Finesse **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Unnatural **Spells:** Caster Level 9

EQUIPMENT

Claw 1: Ghast Claw (1d3 Dmg.; On Monster Hit: Stun) **Claw 2:** Ghast Claw (1d3 Dmg.; On Monster Hit: Stun) **Special Attack:** Ghoul Lord Bite (1d6 Dmg.; On Monster Hit: Stun) **Skin/Hide:** Ghoul/Ghast Prop. (Immunity: Misc.: Critical Hits, Death Magic, Level/Ability Drain, Mind-Affecting Spells, Poison; Turn Resistance +2) **Std. Eqp.:** N/A

GROUP: Mummies

STRATEGY: Mummies can be devastating for an unprepared party. All have the nasty combination of Damage Reduction and partial invulnerability to all three physical attack types. That makes turning and magic the preferred strategies, although figuring out what mummy is immune to what magic type can be confusing. Basic Mummies are vulnerable to fire, but all other mummies are partially immune, and are vulnerable to electricity instead.

Mummy Rot is a particularly debilitating Constitution-sapping disease that should be avoided at all costs. The Greater Mummy and Mummy Lord have divine spellcasting ability, but didn't bother to learn anything good.

SOCIETY: Mummies are the remains of a long-forgotten civilization. They ceaselessly guard their tombs and temples and desire only vengeance against those who would disturb their rest.

Greater Mummy

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 11 | HIT POINTS: 78 | ARMOR CLASS: 18

Their time-shriveled features hidden beneath centuries-old funereal wrappings, these former kings and high priests, now undead, continue to stand guard over their places of rest.

STR	19/4	INT	6/-2	FOR	4
DEX	10/0	WIS	15/2	REF	4
CON	10/0	CHA	15/2	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficency (creature) **Combat Feats:** N/A **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency; Toughness **Magical Feats:** Combat Casting **Other Feats:** Darkvision; Turn Undead **Monster Abilities:** 1 x Aura of Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Warrior/Greater Mummy Slam (2d6 Dmg.; On Monster Hit: Disease Mummy Rot) **Skin/Hide:** Warrior/Greater Mummy Prop. (DR: +2 Soak 10 Dmg.; DV: Electrical 50% Dmg. Vuln.; Immunity: Dmg. Type: Bludgeon, Fire, Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Mummy

RACE: Undead | ALIGNMENT: LE

CHALLENGE RATING: 6 | HIT POINTS: 48 | ARMOR CLASS: 17

Ritually preserved and animated corpses that maintain their timeless vigil over tombs and temple complexes, Mummies are an effective deterrent to would-be grave-robbers.

STR	17/3	INT	6/-2	FOR	2
DEX	8/-1	WIS	14/2	REF	1
CON	10/0	CHA	15/2	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficency (creature) **Combat Feats:** N/A **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Mummy Slam (1d6 Dmg.; On Monster Hit: Disease Mummy Rot) **Skin/Hide:** Mummy Prop. (DR: +1 Soak 5 Dmg.; DV: Fire 50% Dmg. Vuln.; Immunity: Dmg. Type: Bludgeon, Fire, Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Mummy Lord

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 12 | HIT POINTS: 142 | ARMOR CLASS: 18

Their time-shriveled features hidden beneath centuries-old funereal wrappings, these former kings and high priests, now undead, continue to stand guard over their places of rest.

STR	17/3	INT	6/-2	FOR	4
DEX	10/0	WIS	14/2	REF	4
CON	10/0	CHA	15/2	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency; Toughness **Magical Feats:** Combat Casting **Other Feats:** Darkvision; Turn Undead **Monster Abilities:** 3 x Aura of Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Warrior/Greater Mummy Slam (2d6 Dmg.; On Monster Hit: Disease Mummy Rot) **Skin/Hide:** Warrior/Greater Mummy Prop. (DR: +2 Soak 10 Dmg.; DV: Electrical 50% Dmg. Vuln.; Immunity: Dmg. Type: Bludgeon, Fire, Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Warrior Mummy

RACE: Undead | ALIGNMENT: LE

CHALLENGE RATING: 11 | HIT POINTS: 76 | ARMOR CLASS: 18

Their time-shriveled features hidden beneath centuries-old funereal wrappings, these once-heroic warriors and generals now wield their weapons against all who disturb their rest.

STR	19/4	INT	6/-2	FOR	4
DEX	10/0	WIS	14/2	REF	4
CON	10/0	CHA	15/2	WIL	12

SAVES

FEATS & ABILITIES

Active Combat Feats: Disarm; Knockdown; Power Attack **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Cleave; Proficiency (martial, simple) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency; Toughness **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Fear **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Warrior/Greater Mummy Slam (2d6 Dmg.; On Monster Hit: Disease Mummy Rot) **Skin/Hide:** Warrior/Greater Mummy Prop. (DR: +2 Soak 10 Dmg.; DV: Electrical 50% Dmg. Vuln.; Immunity: Dmg. Type: Bludgeon, Fire, Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

GROUP: Others

BAELNORN

(see LICHES)

BODAK

STRATEGY: One word: Death Gaze. Players who don't like instantly dying because they lost a saving throw would do well to avoid Bodaks. Clerics who have it can use Death Ward, and Magic Circle Against Alignment will raise the saving throws of the whole party, but the best defense is vigilance; get away if you can, or slaughter the Bodak immediately. When one appears in a group of mixed Undead, make it your top priority. Its Damage Reduction of 10/+1 shouldn't be a factor, but mages will be flummoxed by its resistances and immunities to Acid, Fire, and Electricity. Ice Storm it is, I guess.

SOCIETY: When you die from a source of pure evil, you become a Bodak. Bodak are pure evil. Ergo, if you're killed by a Bodak's Death Gaze, you become a Bodak in 24 hours. This isn't implemented in the game, of course, but it could be part of a module subplot or something.

CURST

STRATEGY: On offense, Cursts are the same dead as they were alive; the Rogue is deadly from behind. The Ranger is a crack shot. The Monk moves fast and hits hard. On defense, they get a bunch of immunities (including Cold and Fire), Spell Resistance 12, Turn Resistance +10 (ugh!), and Regeneration +1. Fight them as you would any other character of their class, but keep their enhanced defenses in mind.

SOCIETY: It is a difficult process, but a truly twisted evil sorcerer can kill a humanoid in a way that forces the victim into a life of eternal undeath. Cursts often go insane, but otherwise retain their memories and identities. They are not beholden to their creator and if they have the presence of mind to desire anything, it would be someone to cast remove curse and end their suffering.

Any humanoid can be made into a Curst; just give 'em a pale complexion and the Curst Properties as a Skin/Hide.

DOOM KNIGHTS

STRATEGY: If Doom Knights were Paladins in life, how come they have arcane spells? Whatever the reason, they do, and some of them (Fireball and Blade Barrier) can hurt. Thank Buddha virtually all NPC spellcasters waste several slots on See Invisibility, or they'd be even more dangerous!

Anyway, Doom Knights are strong fighters and spellcasters, but they are turnable if you're good enough. They have a Spell Resistance of 12 but no specific immunities outside of the usual undead ones, so spells may be worth trying. Beware of the Greatsword +3 and plan your defenses accordingly.

SOCIETY: Doom Knights are fallen, undead paladins. Rare, but dangerous and phenomenally evil.

LICHES

STRATEGY: Liches are in the same league as Ancient Dragons and defeating one will require a very high-level party. They can cast spells up to 9th level, and specialize in spells that kill players instantly,

as well as the ever-problematic Time Stop. Here's the good news: They lack the natural Spell Resistance of Dragons and Balors, so spellcasters are very useful here. After hitting the party with all your enhancement spells, especially Mass Haste and ones that raise saving throws like Protection from Spells and Magic Circle Against Alignment, Aura Against Alignment and Death Ward, fill your spell slots with Greater Spell Breach, Dismissal, and, if you're feeling really lucky, Feeblemind. It's the spellcaster's job to breach or counter the Lich's defensive spells, Dismiss his summoned creatures, and ensure the fighters have the opening they need to go for the kill.

SOCIETY: Liches are incredibly powerful mages who attained the secrets of cheating death. They are now basically immortal, and will continuously reappear unless you find and destroy the Phylactery that holds their life force.

Most Liches are deeply evil, malevolent beings. But there are a few good Liches, such as the Baelnorn. These elf mages sought eternal undeath to protect their families and kingdoms. They do not have phylacteries, as they were created by a process different from the one that created dark Liches.

MOHRG

STRATEGY: Despite their challenge rating, Mohrgs are really just tough skeletons. They're vulnerable to magic, turning, and all weapons. Their AC is low, they don't have a ton of HP, and they don't do too much damage. Mohrgs pop up frequently in mixed undead encounters, and are much easier to handle than many of their brethren… Kill the Bodaks first, slaughter the Mohrgs, then take down the Revenant.

SOCIETY: Mohrgs are the undead remains of the dregs of humanity: Serial killers, torturers, and sadists. As in life, their only aim is to spread misery and death. The victims of Mohrgs rise again as zombies under a Mohrg's control (this is not implemented in *Neverwinter Nights*, of course).

REVENANT

STRATEGY: Revenants are very skilled fighters, and lower level characters will need to gang up and focus their attacks to overcome the Revenant's Damage Reduction and Regeneration. They are virtually impossible to turn and immune to most spells.

SOCIETY: Revenants are murder victims that have returned from the dead to kill the being that murdered them. They retain their abilities and memories, but tend to lack interest in anything that does not involve their task of vengeance. When a Revenant accomplishes his mission, his spirit moves on. If he fails, he will crumble after six months and die, unavenged. Any unjustly murdered NPC can be made into a Revenant if you give it a suitable complexion and the Revenant Properties hide.

SKELETAL DEVOURER

STRATEGY: Skeletal Devourers are undead Battle Devourers with a bunch of nasty spells. They have Damage Reduction 10/+2 (though Stoneskin could raise that) but no resistance to Fire or Acid spells. If you're going to concentrate on physical attacks, be prepared to overcome an AC of 25. Now might be a good time for that Hero's Strength potion.

SOCIETY: As if Intellect Devourers weren't bad enough, now they're undead. Great. Just frickin' great.

VAMPIRE

STRATEGY: The Vampire Properties list is full of nasty stuff… Damage Reduction, Regeneration, Resistance to Cold, Electricity, Critical Hits, Mind-Affecting Spells, etc… Though there are some clear holes, like vulnerability to Fire, Sonic, and Acid spells. On offense, their most fearsome weapon is their bite, which will drain you two levels per hit. Bring plenty of Restoration spells or potions.

The best way to battle warriors, rogues, and plain-vanilla Vampires is to send familiars, summoned monsters, expendable henchmen, and dominated creatures after them, then stand back and kill the vamps with spells and ranged weapons. So they level-drained your Dire Bear… Who cares? Vampire Priests have mostly defensive and restorative spells, and aren't particularly threatening. But the Vampire Mage list is full of horrible, nasty stuff, so I'd make sure to keep her under a flurry of attacks, even if it means getting close enough to risk level loss.

If you're going up against vampires, always start with a Magic Circle Against Alignment, a 3rd-level spell given to all spell-casting classes except Druids and Rangers. The enhancements are nice, but the protection from mind-control spells is essential. All vampires have some means of dominating characters, and in a small party, a single turncoat henchman can turn the tide of the battle.

SOCIETY: D&D Vampires stick pretty close to the legends. They're repulsed by garlic, mirrors, or holy symbols, they can't enter a home uninvited, they can respawn until their coffin is found and destroyed, and they can't tolerate sunlight. If you're going to make a vampire-oriented campaign, you may consider doing some complicated scripting to add in the vampires' traditional weaknesses. For example:

- Having them respawn every day until their coffin is found and destroyed (the encounter can be set up to "Infinite Respawn" in the toolset, and an onDeath script on the coffin could destroy the encounter).
- Crippling them during daylight (an onHeartbeat script for vampires in outside areas that checks the time of day).

WIGHT

STRATEGY: Getting level drained hurts, and you're unlikely to have access to Restoration magic at the times in which you fight these low-level foes. So let your henchmen, familiars and summoned creatures rush them while you keep your distance and cut them down with area-effect spells and ranged weapons.

SOCIETY: Wights are tribes of undead that hate the living with an unnatural passion. Any humanoid killed by a Wight becomes a Wight that is forced to obey the creature that created it. For this reason, they are among the most feared and despised creatures in the Realms.

Baelnorn

RACE: Undead | ALIGNMENT: NG

CHALLENGE RATING: 19 | HIT POINTS: 110 | ARMOR CLASS: 16

Driven by ambition, some powerful spellcasters attempt to cheat death itself. Those who succeed become Liches. Those who fail are cast into the Abyss.

STR	13/1	INT	12/1	FOR	9
DEX	12/1	WIS	20/5	REF	6
CON	10/0	CHA	14/2	WIL	14

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (morningstar) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency; **Magical Feats:** Combat Casting; Spell Focus (Evocation); Spell Penetrator **Other Feats:** Hardiness vs. Enchantments; Immunity to Sleep; Keen Sense; Skill Affinity (Listen, Search, Spot); Turn Undead; Weapon Proficiency (Elf) **Monster Abilities:** 1 x Aura Of Fear; 3 x Gaze of Doom **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Claw 2:** Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Special Attack:** N/A **Skin/Hide:** Lich Prop. (Ability Bonus: Cha., Int., Wis. +2; DR: +1 Soak 15 Dmg.; Immunity: Dmg. Type: Cold, Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +4) **Std. Eqp.:** N/A

Bodak

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 9 | HIT POINTS: 58 | ARMOR CLASS: 15

With their, gray hairless flesh and empty white gaze, Bodaks are the undead remains of those who have been destroyed by the touch of evil.

STR	13/1	INT	6/-2	FOR	3
DEX	15/2	WIS	12/1	REF	5
CON	10/0	CHA	12/1	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature); Weapon Focus (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** 6 x Gaze, Death **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam 1d8 **Skin/Hide:** Bodak Prop. (DR: +1 Soak 10 Dmg.; Dmg. Resist.: Acid, Fire Resist 20/-; Immunity: Dmg. Type: Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Curst Monk

RACE: Undead | ALIGNMENT: CN

CHALLENGE RATING: 6 | HIT POINTS: 55 | ARMOR CLASS: 16

The eyes of a Curst show pain and madness. Such creatures suffer powerful magical curses that bind them to undeath. They know peace only when destroyed.

STR	12/1	INT	10/0	FOR	6
DEX	17/3	WIS	15/2	REF	7
CON	14/2	CHA	11/0	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: Stunning Fist **Class/Racial Feats:** N/A **Combat Feats:** Improved Unarmed Strike; Weapon Proficiency (monk); Weapon Focus (kama) **Defensive Feats:** Armor Proficiency (light, medium); Deflect Arrows; Dodge; Shield Proficiency; Toughness **Other Feats:** Evasion; Flurry of Blows; Monk AC Bonus; Monk Speed; Purity of Body; Quick To Master; Still Mind **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Curst Prop. (Immunity: Dmg. Type: Cold, Fire 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +1; Spell Resistance Bonus of 12 SR; Turn Resistance +10) **Std. Eqp.:** Kama

Curst Ranger

RACE: Undead | ALIGNMENT: CN

CHALLENGE RATING: 6 | HIT POINTS: 35 | ARMOR CLASS: 18

The eyes of a Curst show pain and madness. Such creatures suffer powerful magical curses that bind them to undeath. They know peace only when destroyed.

STR	14/2	INT	10/0	FOR	3
DEX	20/5	WIS	13/1	REF	6
CON	8/-1	CHA	10/0	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Point Blank Shot; Weapon Proficiency (martial, simple); Weapon Focus (longbow) **Defensive Feats:** Armor Proficiency (light, medium); Shield Proficiency **Other Feats:** Favored Enemy: Giants, Goblinoids; Trackless Step **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Curst Prop. (Immunity: Dmg. Type: Cold, Fire 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +1; Spell Resistance Bonus of 12 SR; Turn Resistance +10) **Std. Eqp.:** Poison Arrow; Longbow

Curst Rogue

RACE: Undead | ALIGNMENT: CN

CHALLENGE RATING: 6 | HIT POINTS: 50 | ARMOR CLASS: 18

The eyes of a Curst show pain and madness. Such creatures suffer powerful magical curses that bind them to undeath. They know peace only when destroyed.

STR	8/-1	INT	10/0	FOR	3
DEX	21/5	WIS	12/1	REF	9
CON	14/2	CHA	10/0	WIL	2

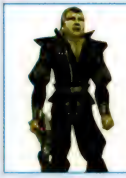
SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency (rogue); Two-Weapon Fighting **Defensive Feats:** Armor Proficiency (light); Dodge **Other Feats:** Evasion; Fearless; Lucky; Skill Affinity (Listen, Move Silently); Sneak Attack (+1d6); Uncanny Dodge 1 **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Curst Prop. (Immunity: Dmg. Type: Cold, Fire 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +1; Spell Resistance Bonus of 12 SR; Turn Resistance +10) **Std. Eqp.:** Short Sword; Short Sword



Curst Warrior

RACE: Undead | ALIGNMENT: CN

CHALLENGE RATING: 6 | HIT POINTS: 43 | ARMOR CLASS: 15

The eyes of a Curst show pain and madness. Such creatures suffer powerful magical curses that bind them to undeath. They know peace only when destroyed.

STR	21/5	INT	6/-2	FOR	5
DEX	14/2	WIS	12/1	REF	3
CON	13/1	CHA	6/-2	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** N/A **Combat Feats:** Weapon Proficiency (martial, simple); Weapon Focus (greataxe) **Defensive Feats:** Armor Proficiency (light, medium); Dodge; Shield Proficiency **Other Feats:** Barbarian Fast Movement; Darkvision; Uncanny Dodge 1, 2 **Monster Abilities:** 1 x Rage **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Curst Prop. (Immunity: Dmg. Type: Cold, Fire 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +1; Spell Resistance Bonus of 12 SR; Turn Resistance +10) **Std. Eqp.:** Greataxe

Doom Knight

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 8 | HIT POINTS: 76 | ARMOR CLASS: 18

The cursed, undead corruptions of once-upright paladins who fell from grace. Doom Knights are the scourge of all lands over which they hold dominion.

STR	21/5	INT	12/1	FOR	10
DEX	15/2	WIS	15/2	REF	8
CON	19/4	CHA	12/1	WIL	8

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Cleave; Weapon Proficiency (martial); Weapon Focus (greatsword) **Defensive Feats:** Armor Proficiency (heavy) **Other Feats:** Darkvision **Monster Abilities:** 2 x Aura of Fear **Spells:** Caster Level 9

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Doom Knight Prop. (DR: +1 Soak 15 Dmg.; Immunity: Misc.: Death Magic, Mind-Affecting Spells, Paralysis; Spell Resistance Bonus of 12 SR) **Std. Eqp.:** Greatsword+3; Half Plate

Doom Knight Commander

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 11 | HIT POINTS: 160 | ARMOR CLASS: 18

The cursed, undead corruptions of once-upright paladins who fell from grace. Doom Knights are the scourge of all lands over which they hold dominion.

STR	16/3	INT	12/1	FOR	10
DEX	15/2	WIS	15/2	REF	8
CON	19/4	CHA	12/1	WIL	8

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Cleave; Weapon Proficiency (martial); Weapon Focus (greatsword) **Defensive Feats:** Armor Proficiency (heavy) **Other Feats:** Darkvision **Monster Abilities:** 2 x Aura of Fear **Spells:** Caster Level 9

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Doom Knight Prop. (DR: +1 Soak 15 Dmg.; Immunity: Misc.: Death Magic, Mind-Affecting Spells, Paralysis; Spell Resistance Bonus of 12 SR) **Std. Eqp.:** Greatsword+3; Half Plate

Lich (1)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 27 | HIT POINTS: 184 | ARMOR CLASS: 17

Driven by ambition, some powerful spellcasters attempt to cheat death itself. Those who succeed become Liches. Those who fail are cast into the Abyss.

STR	10/0	INT	23/6	FOR	7
DEX	14/2	WIS	14/2	REF	8
CON	13/1	CHA	12/1	WIL	14

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (wizard) **Defensive Feats:** Dodge; Great Fortitude; Mobility; Toughness **Magical Feats:** Combat Casting; Empower Spell; Quicken Spell; Spell Focus (Enchantment, Evocation) **Other Feats:** Quick To Master, Summon Familiar **Monster Abilities:** 1 x Aura of Fear; 3 x Gaze, Doom **Spells:** N/A

EQUIPMENT

Claw 1: Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Claw 2:** Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Special Attack:** N/A **Skin/Hide:** Lich Prop. (Ability Bonus: Cha., Int., Wis. +2; DR: +1 Soak 15 Dmg.; Immunity: Dmg. Type: Cold, Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +4) **Std. Eqp.:** N/A

Lich (2)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 18 | HIT POINTS: 112 | ARMOR CLASS: 17

Driven by ambition, some powerful spellcasters attempt to cheat death itself. YThose who succeed become Liches. Those who fail are cast into the Abyss.

STR	10/0	INT	21/5	FOR	5
DEX	14/2	WIS	14/2	REF	6
CON	13/1	CHA	12/1	WIL	10

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (wizard) **Defensive Feats:** Dodge; Toughness **Magical Feats:** Combat Casting; Empower Spell; Quicken Spell; Spell Focus (Evocation) **Other Feats:** Quick To Master; Summon Familiar **Monster Abilities:** 1 x Aura of Fear **Spells:** N/A

EQUIPMENT

Claw 1: Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Claw 2:** Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Special Attack:** N/A **Skin/Hide:** Lich Prop. (Ability Bonus: Cha., Int., Wis. +2; DR: +1 Soak 15 Dmg.; Immunity: Dmg. Type: Cold, Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +4) **Std. Eqp.:** N/A

Lich (3)

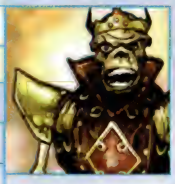
RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 20 | HIT POINTS: 136 | ARMOR CLASS: 10

Driven by ambition, some powerful spellcasters attempt to cheat death itself. Those who succeed become Liches. Those who fail are cast into the Abyss.

STR	10/0	INT	22/6	FOR	6
DEX	11/0	WIS	14/2	REF	5
CON	13/1	CHA	12/1	WIL	12

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (wizard) **Defensive Feats:** Dodge; Toughness **Magical Feats:** Combat Casting; Empower Spell; Quicken Spell; Spell Focus (Evocation); Spell Penetration **Other Feats:** Quick to Master; Summon Familiar **Monster Abilities:** 3 x Aura of Unnatural **Spells:** N/A

EQUIPMENT

Claw 1: Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Claw 2:** Lich Claw (1d8 Dmg.; On Monster Hit: Stun) **Special Attack:** N/A **Skin/Hide:** Lich Prop. (Ability Bonus: Cha., Int., Wis. +2; DR: +1 Soak 15 Dmg.; Immunity: Dmg. Type: Cold, Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +4) **Std. Eqp.:** N/A

Mohrg

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 10 | HIT POINTS: 91 | ARMOR CLASS: 15

The animated corpses of unrepentant villains, Mohrgs are tortured by an all-consuming hatred of living things -- they long only to live again themselves.

STR	21/5	INT	11/0	FOR	4
DEX	13/1	WIS	10/0	REF	5
CON	10/0	CHA	10/0	WIL	9

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge; Mobility **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Slam1d6 **Claw 2:** Slam1d6 **Special Attack:** Mohrg Bite (On Monster Hit: Stun) **Skin/Hide:** Undead Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Revenant

RACE: Undead | ALIGNMENT: N

CHALLENGE RATING: 10 | HIT POINTS: 58 | ARMOR CLASS: 11

STR	20/5	INT	10/0	FOR	6
DEX	13/1	WIS	12/1	REF	3
CON	12/1	CHA	12/1	WIL	3

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Alertness **Combat Feats:** Cleave; Point Blank Shot; Weapon Proficiency (exotic, martial, simple); Weapon Focus (longsword); Weapon Specialization (longsword) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Great Fortitude; Shield Proficiency; Toughness **Other Feats:** Quick To Master **Monster Abilities:** 1 x Gaze, Paralysis **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** N/A **Skin/Hide:** Revenant Prop. (Ability Bonus: Cha. +2, Str. +4; DR: +1 Soak 5 Dmg.; DV: Fire 100% Dmg. Vuln.; Immunity: Dmg. Type: Acid, Cold, Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +3; Turn Resistance +2) **Std. Eqp.:** Longsword +2; Revenant Chainmail +1

Skeletal Devourer

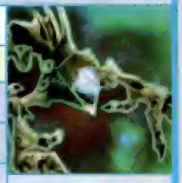
RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 12 | HIT POINTS: 85 | ARMOR CLASS: 25

Returned to vigor by evil sorcerers, these undead intellect Devourers are more resilient and have greater mental acuity than their still-living brethren.

STR	16/3	INT	12/1	FOR	3
DEX	19/4	WIS	12/1	REF	7
CON	10/0	CHA	10/0	WIL	8

SAVES



FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Fear **Spells:** Caster Level 15

EQUIPMENT

Claw 1: Skeletal Devourer Claw (1d8 Dmg.; On Monster Hit: Stun) **Claw 2:** Skeletal Devourer Claw (1d8 Dmg.; On Monster Hit: Stun) **Special Attack:** N/A **Skin/Hide:** Skeletal Devourer Prop. (DR: +2 Soak 10 Dmg.; Immunity: Dmg. Type: Cold, Electrical 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +4) **Std. Eqp.:** N/A

Vampire

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 6 | HIT POINTS: 32 | ARMOR CLASS: 19

STR	22/6	INT	12/1	FOR	1
DEX	17/3	WIS	16/3	REF	4
CON	10/0	CHA	12/1	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (exotic, martial); Weapon Focus (bastard sword); Weapon Specialization (bastard sword) **Defensive Feats:** Dodge; Lightning Reflexes; Mobility **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 5

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Vampire Bite (1d6 Dmg.; On Monster Bite: Level Drain Amount: 2) **Skin/Hide:** Vampire Prop. (DR: +1 Soak 15 Dmg.; Dmg. Resist.: Cold, Electrical 20/-; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +5) **Std. Eqp.:** Bastard Sword

Vampire Mage

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 6 | HIT POINTS: 94 | ARMOR CLASS: 20

Forever anchored to their coffins and the unholy earth of their graves, these nocturnal predators scheme constantly to fill the world with their foul progeny.

STR	16/3	INT	20/5	FOR	4
DEX	18/4	WIS	14/2	REF	7
CON	13/1	CHA	14/2	WIL	9

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (wizard) **Defensive Feats:** Dodge; Lightning Reflexes; Toughness **Magical Feats:** Combat Casting; Empower Spell; Quicken Spell; Spell Focus (Evocation) **Other Feats:** Quick To Master; Summon Familiar **Monster Abilities:** 1 x Gaze, Dominate **Spells:** Wizard Level 5

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Vampire Bite (1d6 Dmg.; On Monster Bite: Level Drain Amount: 2) **Skin/Hide:** Vampire Prop. (DR: +1 Soak 15 Dmg.; Dmg. Resist.: Cold, Electrical 20/-; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +5) **Std. Eqp.:** Dagger +1

Vampire Priest

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 13 | HIT POINTS: 104 | ARMOR CLASS: 17

Forever anchored to their coffins and the unholy earth of their graves, these nocturnal predators scheme constantly to fill the world with their foul progeny.

STR	19/4	INT	12/1	FOR	9
DEX	12/1	WIS	19/4	REF	4
CON	14/2	CHA	16/3	WIL	11

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Weapon Proficiency (simple); Weapon Focus (morningstar) **Defensive Feats:** Armor Proficiency (heavy, light, medium) Dodge; Lightning Reflexes; Shield Proficiency; Toughness **Magical Feats:** Combat Casting; Spell Penetration **Other Feats:** Quick To Master; Turn Undead **Monster Abilities:** 1 x Gaze, Dominate **Spells:** Cleric Level 5

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Vampire Bite (1d6 Dmg.; On Monster Bite: Level Drain Amount: 2) **Skin/Hide:** Vampire Prop. (DR: +1 Soak 15 Dmg.; Dmg. Resist.: Cold, Electrical 20/-; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +5) **Std. Eqp.:** Morningstar

Vampire Rogue

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 12 | HIT POINTS: 104 | ARMOR CLASS: 22

Forever anchored to their coffins and the unholy earth of their graves, these nocturnal predators scheme constantly to fill the world with their foul progeny.

STR	18/4	INT	12/1	FOR	5
DEX	22/6	WIS	14/2	REF	13
CON	14/2	CHA	14/2	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness **Combat Feats:** Ambidexterity; Weapon Proficiency (rogue); Two-Weapon Fighting **Defensive Feats:** Armor Proficiency (light) Dodge; Lightning Reflexes; Toughness **Other Feats:** Evasion; Monster Uncanny Reflex; Quick To Master; Sneak Attack (+1d6); Uncanny Dodge 1, 2 **Monster Abilities:** 1 x Gaze, Dominate

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Vampire Bite (1d6 Dmg.; On Monster Bite: Level Drain Amount: 2) **Skin/Hide:** Vampire Prop. (DR: +1 Soak 15 Dmg.; Dmg. Resist.: Cold, Electrical 20/-; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +5) **Std. Eqp.:** Short Sword +2; Short Sword +2

Vampire Warrior

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 12 | HIT POINTS: 94 | ARMOR CLASS: 19

Forever anchored to their coffins and the unholy earth of their graves, these nocturnal predators scheme constantly to fill the world with their foul progeny.

STR	23/6	INT	12/1	FOR	8
DEX	17/3	WIS	14/2	REF	6
CON	12/1	CHA	14/2	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Att. **Class/Racial Feats:** Alertness **Combat Feats:** Cleave; Improved Critical (bastard sword); Point Blank Shot; Proficiency (exotic, martial, simple); Weapon Focus (bastard sword, longbow); Weapon Specialization (bastard sword) **Defensive Feats:** Armor Proficiency (heavy, light medium); Dodge; Great Fortitude; Lightning Reflexes; Shield Proficiency; Toughness **Other Feats:** Quick To Master **Monster Abilities:** 1 x Gaze, Dominate

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Vampire Bite (1d6 Dmg.; On Monster Bite: Level Drain Amount: 2) **Skin/Hide:** Vampire Prop. (DR: +1 Soak 15 Dmg.; Dmg. Resist.: Cold, Electrical 20/-; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Regeneration +5) **Std. Eqp.:** Bastard Sword +2

Wight

RACE: Undead | ALIGNMENT: LE

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 15

These dark undead seek to destroy all life, filling graveyards with their victims and populating the world with their horrid progeny.

STR	12/1	INT	11/0	FOR	1
DEX	12/1	WIS	13/1	REF	2
CON	10/0	CHA	15/2	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Wight Slam (1d4; On Monster Bite: Level Drain Amount: 1) **Skin/Hide:** Undead Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

GROUP
Shadows & Wraiths

STRATEGY: Only the Shadow Fiend does actual damage; other Shadows prefer to kill by draining Strength. Shadows are very weak, with a meager 5/+1 Damage Reduction, a low AC, few Hit Points, and no immunities to damage dealing spells. So if you can spot them early and get a few bow shots or magic missiles off, you should be able to prevent them from getting close enough to drain a single point.

Allips, Spectres and Wraiths all do minor amounts of damage, and drain different stats. Allips drain Wisdom, Wraiths drain Constitution, and dangerous Spectres suck 2 levels off the top. Spectres and Wraiths are the toughest of the bunch, having a few too many HP to be killable from long range alone. Make sure when they catch up to you they're stuck level draining a Dire Badger or something equally expendable.

SOCIETY: Shadows and Wraiths hate light and will attempt to kill any living creature. Humanoids killed by Shadows or Wraiths return as Shadows and Wraiths themselves. Allips are the remains of the suicidally mad, and directly attack the sanity of their victims. Spectres haunt the place where they die, and eternally resent the living. Like Shadows, Wraiths are beings of darkness that were never actually alive.

Allip

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 26 | ARMOR CLASS: 11

The spectral remains of people driven to madness and suicide. Allips crave only revenge against those who pushed them over the brink.

STR	10/0	INT	11/0	FOR	1
DEX	12/1	WIS	11/0	REF	2
CON	10/0	CHA	18/4	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Allip Claw (1d4 Dmg.; On Monster Hit: Ability Drain Wisdom) **Claw 2:** Allip Claw (1d4 Dmg.; On Monster Hit: Ability Drain Wisdom) **Special Attack:** N/A **Skin/Hide:** Shadow/Wraith Prop. (DR: +1 Soak 5 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +2) **Std. Eqp.:** N/A

Gaseous Form

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 19 | ARMOR CLASS: 12

Shadows are creatures of living darkness, hating life and light with equal fervor. Their touch leaves the chill of nonexistence, making them very dangerous opponents.

STR	10/0	INT	6/-2	FOR	1
DEX	14/2	WIS	12/1	REF	3
CON	10/0	CHA	13/1	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Shadow Claw (On Monster Hit: Ability Drain Strength) **Claw 2:** Shadow Claw (On Monster Hit: Ability Drain Strength) **Special Attack:** N/A **Skin/Hide:** Shadow/Wraith Prop. (DR: +1 Soak 5 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +2) **Std. Eqp.:** N/A

Shadow

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 19 | ARMOR CLASS: 12

Shadows are creatures of living darkness, hating life and light with equal fervor. Their touch leaves the chill of nonexistence, making them very dangerous opponents.

STR	10/0	INT	6/-2	FOR	1
DEX	14/2	WIS	12/1	REF	3
CON	10/0	CHA	13/1	WIL	4

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Shadow Claw (On Monster Hit: Ability Drain Strength) **Claw 2:** Shadow Claw (On Monster Hit: Ability Drain Strength) **Special Attack:** N/A **Skin/Hide:** Shadow/Wraith Prop. (DR: +1 Soak 5 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +2) **Std. Eqp.:** N/A

Shadow Fiend

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 6 | HIT POINTS: 45 | ARMOR CLASS: 16

Dark and dangerous creatures from the dreaded lower planes, Shadow Fiends never hunger or thirst, attacking only to satisfy their desire to do evil.

STR	10/0	INT	12/1	FOR	2
DEX	14/2	WIS	12/1	REF	4
CON	10/0	CHA	13/1	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Dodge **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 14

EQUIPMENT

Claw 1: Claw1d6 **Claw 2:** Claw1d6 **Special Attack:** Bite1d8 **Skin/Hide:** Shadow Fiend Prop. (Immunity: Dmg. Type: Cold, Electrical, Fire 100% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** N/A

Spectre

RACE: Undead | ALIGNMENT: LE

CHALLENGE RATING: 6 | HIT POINTS: 45 | ARMOR CLASS: 13

Spectres are incorporeal undead often mistaken for Ghosts. They haunt the places where they died, retaining their sentience but now hating all living things.

STR	10/0	INT	14/2	FOR	2
DEX	16/3	WIS	14/2	REF	5
CON	10/0	CHA	15/2	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Unnatural **Spells:** N/A

EQUIPMENT

Claw 1: Spectre Claw (On Monster Hit: Level Drain Amount: 2) **Claw 2:** Spectre Claw (On Monster Hit: Level Drain Amount: 2) **Special Attack:** N/A **Skin/Hide:** Shadow/Wraith Prop. (DR: +1 Soak 5 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +2) **Std. Eqp.:** N/A

Wraith

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 4 | HIT POINTS: 32 | ARMOR CLASS: 13

Wraiths are incorporeal creatures born of evil and darkness. They despise all living things, as well as the light that nurtures them.

STR	10/0	INT	14/2	FOR	1
DEX	16/3	WIS	14/2	REF	4
CON	10/0	CHA	15/2	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** N/A **Other Feats:** Darkvision **Monster Abilities:** 1 x Aura of Unnatural **Spells:** N/A

EQUIPMENT

Claw 1: Wraith Claw (On Monster Hit: Ability Drain Constitution) **Claw 2:** Wraith Claw (On Monster Hit: Ability Drain Constitution) **Special Attack:** N/A **Skin/Hide:** Shadow/Wraith Prop. (DR: +1 Soak 5 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Turn Resistance +2) **Std. Eqp.:** N/A

STRATEGY: Skeletons come in all shapes and sizes, but share a few properties. All have 50% immunity to piercing and slashing weapons, so fighters should ditch the blades and use that warhammer they've been lugging around. Isn't it nice being automatically proficient in everything? Most Skeletons are easy to turn but the more powerful ones have Turn Resistance +10, making that strategy ineffective. I guess that's why Clerics also have spells like Searing Light, although even that won't get past the Spell Resistance half the time. On second thought, forget the spells entirely. Your best bet is just bashing them with blunt objects.

SOCIETY: Skeletons have no souls, wills, or agendas. They're just bodies that have been animated by a necromancer, whom they obey mindlessly.

GROUP
Skeletons

Skeleton

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 1/2 | HIT POINTS: 6 | ARMOR CLASS: 10

The bones of the dead, animated by dark magic, Skeletons have no mind of their own. They are but an extension of their master's will.

STR	10/0	INT	10/0	FOR	0
DEX	11/0	WIS	10/0	REF	0
CON	10/0	CHA	11/0	WIL	2

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (martial) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** N/A **Skin/Hide:** Skeleton Prop. (Immunity: Dmg. Type: Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison) **Std. Eqp.:** Battleaxe

Skeleton Chieftain

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 9 | HIT POINTS: 36 | ARMOR CLASS: 15

The bones of the dead, animated by dark magic, Skeletons have no mind of their own. They are but an extension of their master's will.

STR	18/4	INT	10/0	FOR	2
DEX	12/1	WIS	10/0	REF	3
CON	10/0	CHA	11/0	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Prof. (creature) **Combat Feats:** Cleave; Improved Critical (greatsword); Weapon Proficiency (martial, simple); Weapon Specialization (greatsword) **Defensive Feats:** Armor Proficiency (heavy, light, medium) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** N/A **Skin/Hide:** Skeleton Warrior Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Spell Resistance Bonus of 20 SR; Turn Resistance +10) **Std. Eqp.:** Greatsword +1

Skeleton Mage

RACE: Undead | ALIGNMENT: N

CHALLENGE RATING: 4 | HIT POINTS: 14 | ARMOR CLASS: 13

The bones of the dead, animated by dark magic, Skeletons have no mind of their own. They are but an extension of their master's will.

STR	10/0	INT	15/2	FOR	1
DEX	12/1	WIS	10/0	REF	2
CON	10/0	CHA	11/0	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Alertness; Proficiency (creature) **Combat Feats:** Weapon Proficiency (wizard) **Defensive Feats:** N/A **Magical Feats:** Combat Casting **Other Feats:** Darkvision; Summon Familiar **Monster Abilities:** N/A **Spells:** Wizard Level 12

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** N/A **Skin/Hide:** Skeleton Mage/Priest Prop. (Immunity: Dmg. Type: Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Spell Resistance Bonus of 10 SR) **Std. Eqp.:** Dagger +1

Skeleton Priest

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 4 | HIT POINTS: 20 | ARMOR CLASS: 13

The bones of the dead, animated by dark magic, Skeletons have no mind of their own. They are but an extension of their master's will.

STR	10/0	INT	10/0	FOR	1
DEX	12/1	WIS	15/2	REF	2
CON	10/0	CHA	11/0	WIL	7

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Prof. (creature) **Combat Feats:** Weapon Proficiency (simple) **Defensive Feats:** Armor Proficiency (heavy, light, medium); Shield Proficiency **Magical Feats:** Combat Casting **Other Feats:** Darkvision; Turn Undead **Monster Abilities:** N/A **Spells:** Cleric Level 2

EQUIPMENT

Claw 1: Claw1d4 **Claw 2:** Claw1d4 **Special Attack:** N/A **Skin/Hide:** Skeleton Mage/Priest Prop. (Immunity: Dmg. Type: Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Spell Resistance Bonus of 10 SR) **Std. Eqp.:** Mace +1

Skeleton Warrior (1)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 8 | HIT POINTS: 39 | ARMOR CLASS: 15

The bones of the dead, animated by dark magic, Skeletons have no mind of their own. They are but an extension of their master's will.

STR	18/4	INT	10/0	FOR	2
DEX	12/1	WIS	10/0	REF	3
CON	10/0	CHA	11/0	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial); Weapon Specialization (greatsword) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** N/A **Skin/Hide:** Skeleton Warrior Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Spell Resistance Bonus of 20 SR; Turn Resistance +10) **Std. Eqp.:** Greatsword +1

Skeleton Warrior (2)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 8 | HIT POINTS: 39 | ARMOR CLASS: 15

The bones of the dead, animated by dark magic, Skeletons have no mind of their own. They are but an extension of their master's will.

STR	18/4	INT	10/0	FOR	2
DEX	12/1	WIS	10/0	REF	3
CON	10/0	CHA	11/0	WIL	5

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial); Weapon Specialization (greataxe) **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: Claw1d8 **Claw 2:** Claw1d8 **Special Attack:** N/A **Skin/Hide:** Skeleton Warrior Prop. (DR: +1 Soak 5 Dmg.; Immunity: Dmg. Type: Pierce, Slash 50% Imm. Bonus; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Spell Resistance Bonus of 20 SR; Turn Resistance +10) **Std. Eqp.:** Greataxe+1

STRATEGY: Finally, something clerics can just turn and blast with sunlight that doesn't have any fancy resistances. Zombies are slow and dumb and easy to kill with just about anything. Spells? Anything that deals damage is fine. Bladed weapons? Go for it.

The only Zombie with a special property is the Tyrantfog Zombie, with his Constitution sapping Mist and Strength draining touch. Unless you can turn them all immediately, you'll be needing a Lesser Restoration or a good rest after a fight with a pack of those guys.

SOCIETY: Zombies are mindless, reanimated corpses. That evil Xvim guy, of "Hound of Xvim" fame, made a killing mist to wipe out the followers of a rival god. Its victims now spread the plague as Tyrantfog Zombies.

GROUP
Zombies

Tyrantfog Zombie

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 3 | HIT POINTS: 30 | ARMOR CLASS: 10

The remains of priests and worshipers of evil deities, the rotting corpses of these Zombies exude a sickening green fog.

				SAVES		
STR	13/1	INT	6/-2		FOR	1
DEX	10/0	WIS	12/1		REF	4
CON	10/0	CHA	13/1		WIL	6

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Weapon Proficiency (simple); Weapon Specialization (mace) **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** 1 x Tyrant Fog Zombie Mist **Spells:** N/A

EQUIPMENT

Claw 1: Zombie, Tyrantfog Claw (1d6 Dmg.; On Monster Hit: Disease Zombie Creep) **Claw 2:** Zombie, Tyrantfog Claw (1d6 Dmg.; On Monster Hit: Disease Zombie Creep) **Special Attack:** N/A **Skin/Hide:** Zombie, Tyrantfog Prop. (DR: +1 Soak 15 Dmg.; Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Special Walk: Zombie Walk) **Std. Eqp.:** Mace

Zombie (1)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 1 | HIT POINTS: 15 | ARMOR CLASS: 9

Corpses reanimated through necromancy, Zombies are mindless undead that shamble about, trying to satiate the dull ache of hunger in their decaying gut.

				SAVES		
STR	13/1	INT	10/0		FOR	0
DEX	8/-1	WIS	10/0		REF	0
CON	10/0	CHA	3/-4		WIL	3

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam1d6 **Skin/Hide:** Zombie Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Special Walk: Zombie Walk) **Std. Eqp.:** N/A

Zombie (2)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 1 | HIT POINTS: 15 | ARMOR CLASS: 9

Corpses reanimated through necromancy, Zombies are mindless undead that shamble about, trying to satiate the dull ache of hunger in their decaying gut.

				SAVES		
STR	13/1	INT	10/0		FOR	0
DEX	8/-1	WIS	10/0		REF	0
CON	10/0	CHA	3/-4		WIL	3

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** N/A **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Slam1d6 **Skin/Hide:** Zombie Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Special Walk: Zombie Walk) **Std. Eqp.:** N/A

Zombie Lord

RACE: Undead | ALIGNMENT: CE

CHALLENGE RATING: 6 | HIT POINTS: 129 | ARMOR CLASS: 10

STR	13/1	INT	10/0	FOR	3
DEX	8/-1	WIS	10/0	REF	2
CON	10/0	CHA	10/0	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: N/A **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Ambidexterity; improved Two-Weapon Fighting; Weapon Proficiency (exotic); Two-Weapon Fighting **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** Caster Level 9

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Zombie Lord Slam **Skin/Hide:** Zombie Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Special Walk: Zombie Walk) **Std. Eqp.:** Kama +1; Kama +1

Zombie Warrior (1)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 33 | ARMOR CLASS: 12

Corpses reanimated through necromancy, Zombies are mindless undead that shamble about, trying to satiate the dull ache of hunger in their decaying gut.

STR	17/3	INT	10/0	FOR	1
DEX	8/-1	WIS	10/0	REF	0
CON	10/0	CHA	3/-4	WIL	6

SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial, simple) **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Zombie Warrior Slam **Skin/Hide:** Zombie Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Special Walk: Zombie Walk) **Std. Eqp.:** Club

Zombie Warrior (2)

RACE: Undead | ALIGNMENT: NE

CHALLENGE RATING: 3 | HIT POINTS: 33 | ARMOR CLASS: 12

Corpses reanimated through necromancy, Zombies are mindless undead that shamble about, trying to satiate the dull ache of hunger in their decaying gut.

STR	17/3	INT	10/0	FOR	1
DEX	8/-1	WIS	10/0	REF	0
CON	10/0	CHA	3/-4	WIL	6

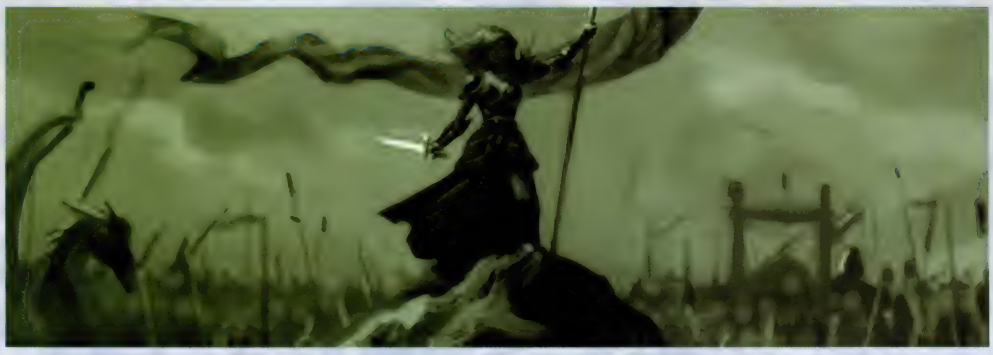
SAVES

FEATS & ABILITIES

Active Combat Feats: Power Attack **Class/Racial Feats:** Proficiency (creature) **Combat Feats:** Cleave; Weapon Proficiency (martial, simple) **Defensive Feats:** Toughness **Other Feats:** Darkvision **Monster Abilities:** N/A **Spells:** N/A

EQUIPMENT

Claw 1: N/A **Claw 2:** N/A **Special Attack:** Zombie Warrior Slam **Skin/Hide:** Zombie Prop. (Immunity: Misc.: Critical Hits, Death Magic, Disease, Level/Ability Drain, Mind-Affecting Spells, Paralysis, Poison; Special Walk: Zombie Walk) **Std. Eqp.:** Club



Races

Dwarves

DESCRIPTION

ABILITY ADJUSTMENTS: +2 Con, -2 Cha
FAVORED CLASS: Fighter

Dwarves get a bunch of interesting bonus feats, including three Battle Training feats (similar to Rangers' "Favored Enemy") and the invaluable Hardiness vs. Spells. The stat adjustments make it hard to play a Charisma-based class (Bard, Sorcerer, Paladin), but outside of that, Dwarves can excel at any class.

BONUS FEATS

Battle Training vs. Giants
Use: Automatic
This feat grants a +4 dodge bonus to AC against giants.

Battle Training vs. Goblins
Use: Automatic
This feat grants a +1 racial bonus on attack rolls against goblins and bugbears.

Battle Training vs. Orcs
Use: Automatic
This feat grants a +1 racial bonus on attack rolls against orcs.

Darkvision
Use: Automatic
This feat grants the ability to see in the dark.

Hardiness vs. Poisons
Use: Automatic
This feat grants a +2 racial bonus on saving throws vs. poison.

Hardiness vs. Spells
Use: Automatic
This feat grants a +2 racial bonus on saving throws vs. spells.

Skill Affinity: Lore
Use: Automatic
This feat grants a +2 racial bonus to Lore checks.

Stonecunning
Use: Automatic
This feat grants a +2 bonus to Search checks made in interior areas.

Elves

DESCRIPTION

ABILITY ADJUSTMENTS: +2 Dex, -2 Con
FAVORED CLASS: Wizard

The favored class of elves may be Wizard, but they make amazing rogues. Keen Sense allows you to play through the game without the speed penalty associated with Search mode, the Dex bonus allows you to crank your primary stat to 18 or 20 without paying too much for it, the Skill Affinities basically give you Alertness for free, and Weapon Proficiency: Elf grants elves the ability to use otherwise-prohibited longswords and longbows. These are also strong benefits for any spellcaster or ranger, and elves should only be avoided when making tank-type characters that need a good Con score.

BONUS FEATS

Hardiness vs. Enchantments
Use: Automatic
This feat grants a +2 bonus on saving throws against mind-affecting spells.

Immunity to Sleep
Use: Automatic
This feat grants immunity to all Sleep spells and effects.

Keen Sense
Use: Automatic
Characters with this feat may apply their full Search skill even when making a passive search.

Low-Light Vision
Use: Automatic
This feat grants the ability to see in the dark, but not as far as with Darkvision.

Skill Affinity: Listen
Use: Automatic
This feat grants a +2 bonus to Listen checks.

Skill Affinity: Search
Use: Automatic
This feat grants a +2 bonus to Search checks.

Skill Affinity: Spot
Use: Automatic
This feat grants a +2 bonus to Spot checks.

Weapon Proficiency: Elf
Use: Automatic
This feat allows the character to use the longsword, rapier, longbow, and shortbow.

Gnomes

DESCRIPTION

ABILITY ADJUSTMENTS: +2 Con, -2 Str
FAVORED CLASS: Wizard

Non-spellcasters have little to gain from the Gnome selection of bonus feats. But mages will definitely appreciate the Con bonus, the Spell Focus (Illusion), and the Skill Affinity of Concentration. Due to their smaller size, Gnomes cannot wield large weapons (and must wield medium weapons two-handed), making any Fighter class a practical impossibility.

BONUS FEATS

Battle Training vs. Giants
Use: Automatic
This feat grants a +4 dodge bonus to AC against giants.

Battle Training vs. Goblins
Use: Automatic
This feat grants a +1 racial bonus on attack rolls against goblins and bugbears.

Battle Training vs. Reptilians
Use: Automatic
This feat grants a +2 racial bonus on attack rolls against reptilian humanoids.

Hardiness vs. Illusions
Use: Automatic
This feat grants a +2 racial bonus on saving throws against mind-affecting spells.

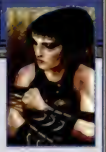
Low-light Vision
Use: Automatic
This feat grants the ability to see in the dark, but not as far as with Darkvision.

Skill Affinity: Concentration
Use: Automatic
This feat grants a +2 bonus to Concentration checks.

Skill Affinity: Spot
Use: Automatic
This feat grants a +2 bonus to Listen checks.

Spell Focus: Illusion
Use: Automatic
A character with this feat is adept in Illusion-school magic spells, gaining a +2 bonus to spell save DC for all spells that they cast from that school.

Halflings



DESCRIPTION

ABILITY ADJUSTMENTS: +2 Dex, -2 Str.
FAVORED CLASS: ROGUE

Halflings are ideal as rogues, and fairly weak at just about everything else... Although wizards, bards, and sorcerers could make good use of the halfling's Dexterity Bonus, luck, and prowess with throwing weapons like darts and shuriken. Since halflings share the small stature of gnomes and cannot use large weapons (and must use medium weapons two-handed), any other class would be a rough fit.

BONUS FEATS

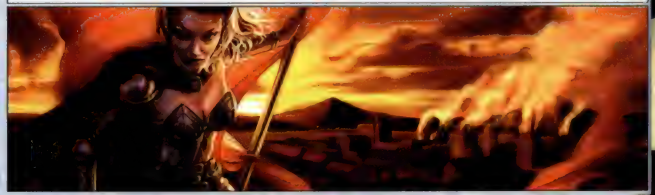
Fearless
Use: Automatic
This feat grants a +2 bonus on saving throws against Fear spells and effects.

Good Aim
Use: Automatic
This feat grants a +1 attack bonus on attacks made with throwing weapons.

Lucky
Use: Automatic
This feat grants a +1 bonus to all saving throws.

Skill Affinity: Listen
Use: Automatic
This feat grants a +2 bonus to Listen checks.

Skill Affinity: Move Silently
Use: Automatic
This feat grants a +2 bonus to Move Silently checks.



STATISTICS, ANALYSIS, & BONUS FEAT LISTINGS FOR EVERY RACE!

Bard

Hit Dice: d6	**Skill Points:** 4
Alignments: Any Nonlawful	**Weapons:** All Simple Weapons
Armor: Light and Med.	**Shields:** Yes

CLASS | Bard

ANALYSIS

Bards are one of the most interesting and versatile classes available under 3rd Edition rules. With their d6 hit dice they're tougher than any other arcane spell caster, and their new and improved Bard Song makes them a welcome addition to any party.

The Bard's spell list emphasizes defensive, healing, summoning, and enhancement spells, which tend to be more useful in larger parties than when playing solo. They get a decent amount of skill points and only Rogues get more class skills.

Unfortunately, Neverwinter Nights does not use the Bards-get-one-free-bonus-weapon rule in 3rd Edition, so Bards are restricted to simple weapons. They have the same attack bonuses as Clerics and Rogues, so they're much better than Sorcerers and Wizards in combat. Bards can wear armor and use shields, but you might not want to because of the risk of spell failure. Bards advance in spells slowly, and don't get many offensive spell options. However, they can use most wands and scrolls naturally, and since Use Magic Device is a class skill, they can end up using all sorts of items restricted to other classes.

With their party-pumping Bard Song, utility spells, and unbeatable Lore, Bards are a great utility character that can be an asset to any large party. But as a solo character, Bards are a challenge. They're a little too weak in combat, and their limited spell list and slow spell progression don't really make up for it. Their greatest strength, Bardic Music, isn't as useful when playing alone.

LEVEL PROGRESSION CHART

Level	Base Attack Bonus	Fort Save	Ref Save	Will Save	Bardic Music Effects
1	+0	+0	+2	+2	+1 to Attack and Damage rolls (required: 1 Perform)
2	+1	+0	+3	+3	+1 to Will saves (required: 6 Perform)
3	+2	+1	+3	+3	+1 to Damage rolls and +1 to Fortitude saves (required: 9 Perform)
4	+3	+1	+4	+4	
5	+3	+1	+4	+4	
6	+4	+2	+5	+5	+1 to Reflex saves (required: 12 Perform)
7	+5	+2	+5	+5	
8	+6/+1	+2	+6	+6	+1 to Attack rolls, +8 temporary HP (required: 15 perform)
9	+6/+1	+3	+6	+6	
10	+7/+2	+3	+7	+7	
11	+8/+3	+3	+7	+7	+2 to Dodge Armor Class (required: 18 Perform)
12	+9/+4	+4	+8	+8	
13	+9/+4	+4	+8	+8	
14	+10/+5	+4	+9	+9	+1 to Damage rolls, +8 temporary HP, +1 Dodge Armor Class (required: 21 Perform)
15	+11/+6/+1	+5	+9	+9	+1 Will saves, +1 Reflex saves, +1 Fortitude saves, +1 Dodge Armor Class (required: 24 Perform)
16	+12/+7/+2	+5	+10	+10	+1 Will saves, +4 temporary HP, +1 Dodge Armor Class (required: 25 Perform) *
17	+12/+7/+2	+5	+10	+10	
18	+13/+8/+3	+6	+11	+11	
19	+14/+9/+4	+6	+11	+11	
20	+15/+10/+5	+6	+12	+12	

* For each additional 5 Perform and 1 class level an additional +2 temporary HP are granted.

SUGGESTIONS

Charisma is the basis of the Bard's spell-casting abilities and their best class skills: Perform, Persuade, and Use Magic Device. You'll probably want a 16. Dexterity is nearly as important, since you'll need the AC bonus and will want to use primarily long range weapons. Intelligence is important because Bards need skills badly. Bardic Music may be free, but you have to pay for Perform every level. Constitution is crucial for staying alive, keeping your spells from being disrupted, and compensating for the Bard's sub-par Fortitude saves. Jack up those 4 stats and leave Wisdom and Strength at 8.

When it comes to spells, you won't get a lot of options. Always take the Summon Monster spells, because the key to solo Bard strategy is hiding behind your dire beasts and using long-range weapons. Other good low-level spells include Sleep, Ghostly Visage (a must), Hold Person, Haste, and Invisibility Sphere. If you're going to focus on long-range weapons, take feats in that area, such as Point Blank Shot and Rapid Shot. Weapon Finesse is tempting, but melee combat is for fools, I say, fools! Concentration and Combat Casting aren't as important as they are for other spell-casters; if you're casting Bard spells during combat (instead of before or after), you're doing something wrong!

BASE SPELLS PER DAY

Level	0	1	2	3	4	5	6
1	2						
2	3	0					
3	3	1					
4	3	2	0				
5	3	3	1				
6	3	3	2				
7	3	3	2	0			
8	3	3	3	1			
9	3	3	3	2			
10	3	3	3	2	0		
11	3	3	3	3	1		
12	3	3	3	3	2		
13	3	3	3	3	2	0	
14	4	3	3	3	3	1	
15	4	4	3	3	3	2	
16	4	4	4	3	3	2	0
17	4	4	4	4	3	3	1
18	4	4	4	4	4	3	2
19	4	4	4	4	4	4	3
20	4	4	4	4	4	4	4

BASE SPELLS KNOWN

Level	0	1	2	3	4	5	6
1	4						
2	5	2					
3	6	3					
4	6	3	2				
5	6	4	3				
6	6	4	3				
7	6	4	4	2			
8	6	4	4	3			
9	6	4	4	3			
10	6	4	4	4	2		
11	6	4	4	4	3		
12	6	4	4	4	3		
13	6	4	4	4	4	2	
14	6	4	4	4	4	3	
15	6	4	4	4	4	3	
16	6	5	4	4	4	4	2
17	6	5	5	4	4	4	3
18	6	5	5	5	4	4	3
19	6	5	5	5	5	4	4
20	6	5	5	5	5	5	4

ABILITIES

BARDIC KNOWLEDGE
Gained: 1st level
Used: Automatic
Bards add their class level to their Lore skill. Now you can free up the skill points you would have put into Lore and pump 'em into Perform.

BARDIC MUSIC
Gained: 1st level
Used: Automatic
Bardic Music is one of the few things in Neverwinter Nights that works drastically different than it did in 3rd Edition. Surprisingly, it's even better now; Bards can still perform while they fight, and no musical instrument is required. It affects all allies within 30 feet, and Bards can perform once per day, per level. The song lasts for 6 rounds no matter what you do (including casting spells). You also need to have a requisite score in Perform to get each effect.
Instead of choosing a specific song, you get the bonuses listed on the chart above. Effects are cumulative, so at level 8, if you have the requisite 15 in your Perform skill (including Charisma bonus) you get +2 to Attack, +2 to damage, +1 to each of your saves, and +8 temporary hit points.

SPELL LISTS

0-Level Bard Spells
In large parties, Daze can be useful early in the game, stunning attackers that would otherwise overwhelm your tank character and setting him up for easy hits. Note that if you're playing on a module that uses true D&D Death and Dying rules (they're not in the official campaign, but the functionality is programmed into the game), Cure Minor Wounds is invaluable for stabilizing dying characters.

- **Cure Minor Wounds**
- **Daze**
- **Light**

1st-Level Bard Spells
Bards have some good stuff here. Summon Creature I is a must-pick spell, giving Bards something to hide behind and another source of damage. Grease is a useful area effect spell if you and your party are using ranged weapons and can avoid falling into it yourself. Sleep is great early in the game (but weak later), while Charm Person is less powerful early on but more valuable later. Protection from Alignment is a great pre-boss enhancement spell. Identify is a waste with Bardic Lore, so save the spell slot for something better.

- **Charm Person**
- **Cure Light Wounds**
- **Grease**
- **Identify**
- **Lesser Dispel**
- **Mage Armor**
- **Protection from Alignment**
- **Scare**
- **Sleep**
- **Summon Creature I**

2nd-Level Bard Spells
Summon Creature II and Ghostly Visage are the standouts at level 2, but your third pick spell depends on your play style. If you're doing a lot of Dexterity-dependent rogue stuff, like picking locks, Cat's Grace can be a nice way to boost your skill checks a point or two and raise your AC at the same time. Invisibility is fun, but Bards have few ways of capitalizing on it. Hold Person can be fantastic against bosses. Sound Burst is a rare bardic damage spell, but doesn't grow in strength and becomes a wasted spell slot later in the game.

- **Blindness/Deafness**
- **Bull's Strength**
- **Cat's Grace**
- **Clarity**
- **Cure Moderate Wounds**
- **Darkness**
- **Eagle's Splendor**
- **Fox's Cunning**
- **Ghostly Visage**
- **Hold Person**
- **Invisibility**
- **Owl's Wisdom**
- **See Invisibility**
- **Silence**
- **Sound Burst**
- **Summon Creature II**
- **Ultravision**

(continued next page)

ANALYSIS, SUGGESTIONS, LEVEL PROGRESSION CHARTS, AND STATISTICS FOR ALL CHARACTER CLASSES!

Chapter 1: Part 1
City Core

1 The Hall of Justice

After the Prelude (no need to cover that here), your quest begins in the Temple of Tyr, also known as the Hall of Justice. This will be your base of operations in Neverwinter. Talk with Fenthick and Desther (always an unpleasant experience), and continue into the next area to speak with Aribeth. Lovely Ari will be your source of updates for Chapter 1's main quest (***"The Wailing and the Waterdhavians"***), free healing and restoration, and clerical supplies. After a quick update, she'll give you the **Stone of Recall**, a magic item that will teleport you back to this temple at any time. You can then hop right back into the portal and end up right where you were when you used the stone. This is a very powerful effect, since it allows you to warp out in the middle of battles, get healed by Aribeth, and then go right back to the action.

The other important character here is Sergol, the wizard who will be watching over you and respawning you when you die. Using his services will carry a hefty XP and gold penalty, so don't do it often.

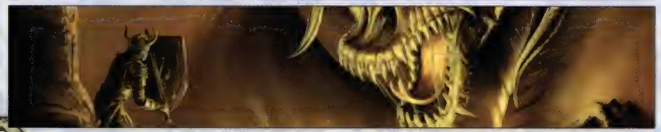
As you exit the Hall, you'll pass Tomi Undergallows, a rogue willing to offer his service as a henchman for 150 bucks (or 125 if you're charismatic enough to bargain him down). Tomi will come in handy often, but there are other henchmen at the Trade of Blades building outside, and a level 3 character may want someone sturdier than the frail halfling for their first few quests.

Class Quest 1: The Search for Never's Tomb

If you continue north through the Hall instead of heading west into town, you'll end up in the apse of Reverend Judge Oleff. He'll start by giving you some background on the Tyrran faith, but the conversation will soon turn to the topic of artifact recovery. If he senses you're "spiritually minded" (a cleric, paladin, or monk), he will enlist your help in a side quest, ***"The Search for Never's Tomb."*** If you're not a member of one of those classes, you can convince him that you might still be of use by passing a Persuade check (DC 24).

If you fail, you can always try again when your Persuade skill is higher.

The first stop on this quest is in the Peninsula District, where you'll deliver the **Letter from Oleff** and learn more about where to find the three artifacts. For each artifact you return, Oleff will give you 200 gold. If you get all three artifacts, he'll throw in 500 XP and an **Amulet of Natural Armor +1**. Succeed at a Persuade check, and he'll give you **Adamantite**, which you can use to forge a powerful weapon at the Shining Knight.

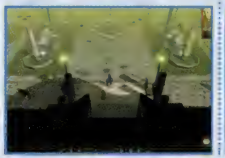


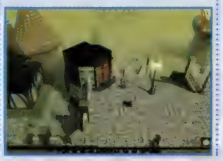
2 Neverwinter City Core

As you leave the Hall of Justice, you'll be approached by a girl named Bethany, who will beseech you to help quell a prison riot in the Peninsula District.

Each of Neverwinter's other four districts has a quest based on one of the missing Waterdhavian creatures, and you can do them in any order you want. Whether the Peninsula District is truly the easiest is debatable, but we have to start somewhere, and can you really say no to poor little Bethany?

There's no hurry, of course. Before you rush off to the Peninsula District, you'll probably want to do some shopping, hire a henchman, enlist in any number of class-specific quests, and raid the unguarded treasure chests throughout the City Core. If you're really greedy, and can pick a lock (DC 22) or smash down a solid door, you can raid the storage rooms in the guard towers on the upper level of the City Core. Henchmen like lock-picking Tomi and smash-happy Daelan can help with the doors.





A Class Quest 2: Cloaktower Membership

Sorcerers and Wizards can apply for membership at the Many-Starred Cloak Guild in the Cloaktower (this is the only class quest that other classes can *not* get with a successful Persuade check). Before they'll let you in, you have to take part in a little scavenger hunt. You'll find arcane laboratories throughout Neverwinter's four other districts, all subtly marked with a sundial. **Eltoora's Key** (provided here) will give you access to these rooms, and if you retrieve the **Rare Earth Clay**, **Flask of Water**, **Puff of Fog**, and **Kindling Wood**, in that order, you're in… For the first part, at least. The final challenge of the ***Cloaktower Membership*** quest will be discussed in our section on the Docks District.

B Class Quest 3: Zoo Rescue

A Druid named Nyatar is standing by the large tree south of the Hall of Justice. Nyatar offers a shop that specializes in druid supplies, but of more interest is the ***Zoo Rescue*** quest he'll add to your journal. If you're a Druid or Ranger, he'll ask you to help him right off, but members of any other class will need to succeed on a Persuade check (DC 24) to get in on the action. The quest is to free the animals from the dismal Blacklake Zoo. To do so, he'll give you a **Zoo Key Copy** and **Teleport via Plants Scroll**, and ask you to use it on the zoo's tree (only one

COMPLETE WALKTHROUGH, WITH RENDERED MAPS, QUEST COVERAGE, NPC STATISTICS, MONSTER STRATEGIES, AND LOCATIONS OF HIDDEN TREASURE!

Feats

Alertness

DESCRIPTION

Type of Feat: General.
Prerequisite: None.
Specifics: +2 bonus to Spot and Listen checks due to finely tuned senses.
Use: Automatic.

Ambidexterity

DESCRIPTION

Type of Feat: General.
Prerequisite: Dex 15+.
Required for: Improved Two-Weapon Fighting.
Specifics: When two-weapon fighting, this feat reduces the penalty of the off-hand weapon by 4.
Use: Used automatically when two-weapon fighting. Rangers receive this feat for free at level 1.

Armor Proficiency Heavy

DESCRIPTION

Type of Feat: General.
Prerequisite: Armor Proficiency (Light) and Armor Proficiency (Medium).
Specifics: This feat grants the knowledge to make effective use of heavy armor.
Use: Automatic. Fighters, paladins, and clerics receive this feat for free.

Armor Proficiency Medium

DESCRIPTION

Type of Feat: General.
Prerequisite: Armor Proficiency (Light).
Required for: Armor Proficiency (Heavy).
Specifics: This feat grants the knowledge to make effective use of medium armor.
Use: Automatic. All classes except monks, rogues, sorcerers, and wizards have this feat for free.

Armor Proficiency Light

DESCRIPTION

Type of Feat: General.
Prerequisite: None.
Required for: Armor Proficiency (Medium).
Specifics: This feat grants the knowledge to make effective use of light armor.
Use: Automatic. All classes except monks, sorcerers, and wizards have this feat for free.

Called Shot

DESCRIPTION

Type of Feat: General.
Prerequisite: Base attack bonus of +1 or higher.
Required for: Sap.
Specifics: Grants the ability to make a potentially disabling attack against an opponent's arms or legs. Called shots are made at a -4 penalty. Called shots must overcome the target's Discipline skill check, and if successful will damage the target based on the location of the called shot. A called shot against the legs will reduce the opponent's movement rate by 20% and give them a -2 cumulative penalty to their Dexterity. A called shot against the arms will apply a cumulative -2 penalty to the creature's attack rolls. Successful called shots last for four rounds.
Use: Selected

Cleave

DESCRIPTION

Type of Feat: General.
Prerequisite: Str 13+, Power Attack feat.
Specifics: If a character with this feat kills an opponent in melee combat, he gets a free attack against any opponent who is within melee attack range.
Use: Automatic.

Combat Casting

DESCRIPTION

Type of Feat: General.
Prerequisite: Ability to cast 1st-level spells.
Specifics: Character is adept at casting spells in combat, removing the standard -4 penalty to Concentration checks when within three meters of an enemy.
Use: Automatic.

Deflect Arrows

DESCRIPTION

Type of Feat: General.
Prerequisite: Dex 13+, Improved Unarmed Strike.
Specifics: Character can attempt to deflect one incoming missile attack per round (Reflex save made against DC 20).
Use: Automatic if not caught flat-footed. Monks receive this feat for free at level 2.

Disarm

DESCRIPTION

Type of Feat: General.
Prerequisite: Int 13+.
Requires for: Improved Disarm.
Specifics: The character can attempt to disarm an opponent in melee combat. Attempting a disarm applies a -6 penalty to the character's attack roll, and the combatant with the larger weapon gains a +4 bonus per size category of difference. A successful hit deals normal damage, and if the opponent fails a Discipline check then the weapon flies from the opponent's hands.
Use: Selected. A disarm attempt provokes an attack of opportunity.

Dodge

DESCRIPTION

Type of Feat: General.
Prerequisite: Dex 13+
Requires for: Mobility
Specifics: Increased agility grants a +1 dodge bonus to AC against attacks from a character's current target (or last attacker).
Use: Automatic, though a condition that negates a Dexterity bonus to AC also negates any dodge bonuses. Multiple dodge bonuses (different feats, racial bonuses) are cumulative.

Empower Spell

DESCRIPTION

Type of Feat: Metamagic.
Prerequisite: Ability to cast 2nd-level spells.
Specifics: Magical expertise allows certain spells to be cast with a 50% increase in variable numeric effects (number of targets, damage, etc), excluding duration.
Use: Empowered spells occupy spell slots two levels higher than normal. Saving throws and opposed rolls, such as those made when dispel magic is cast, are not affected.

Extend Spell

DESCRIPTION

Type of Feat: Metamagic.
Prerequisite: Ability to cast 1st-level spells.
Specifics: Magical expertise allows certain spells to be cast with a 100% increase in duration.
Use: Extended spells occupy spell slots one level higher than normal. Spells with an instantaneous or permanent duration are not affected.

Extra Turning

DESCRIPTION

Type of Feat: Special.
Prerequisite: Exclusive to cleric or paladin.
Specifics: This divine ability allows the character to turn undead six additional times per day.
Use: Automatic.

Great Fortitude

DESCRIPTION

Type of Feat: General.
Prerequisite: None.
Specifics: A character with this feat is very hardy, gaining a +2 bonus to all Fortitude saving throws.
Use: Automatic.

Improved Critical

DESCRIPTION

Type of Feat: General.
Prerequisite: Proficiency with the chosen weapon, base attack bonus +8 or higher.
Specifics: Combat ability doubles the critical threat range with a given weapon. A longsword that normally threatens a critical on a roll of 19-20 would now threaten a critical on a roll of 17-20.
Use: Automatic. The threat range of a Keen weapon is already doubled, increasing to triple with this feat. This feat can be selected multiple times, applying to a new weapon category each time.

Improved Disarm

DESCRIPTION

Type of Feat: General.
Prerequisite: Int 13+, Disarm.
Specifics: A character with this feat has learned not to provoke an attack of opportunity when attempting a disarm, and has no risk of being disarmed himself if he fails in his attempt. Success knocks the opponent's weapon away from them. Success while unarmed takes the weapon from them. The penalty to make a disarm attempt is reduced to -4.
Use: Selected.

Improved Knockdown

DESCRIPTION

Type of Feat: General.
Prerequisite: Knockdown, base attack bonus +7.
Specifics: Characters with this feat have learned to use the Knockdown ability as if one size category larger than they really are. All other Knockdown conditions still apply.
Use: Selected. Monks receive this feat for free at 6th level.

DETAILED EXPLANATIONS FOR ALL FEATS!

Skills

Animal Empathy

DESCRIPTION

Ability: Charisma
Classes: Druid and Ranger
Requires Training: Yes
A successful check allows a character to charm or dominate certain hostile creatures.
Check: Animals and dire animals have a DC of 20 + the creature's hit die. For beasts and magical Beasts, the DC is 24 + the creature's hit die. If the check succeeds, the creature is charmed, or, if the check exceeds the DC by six or greater, the creature is dominated.
Special: If the character fails their check by five or more, the creature will go hostile.
Use: Select this skill and then select the target creature. Dominated creatures will be commanded as henchmen. The creature will remain charmed or dominated for one round per level of the character using the skill.

Concentration

DESCRIPTION

Ability: Constitution
Classes: All
Requires Training: No
Concentration checks are made whenever a character is distracted during the act of casting a spell. It is also used to avoid the effects of the Taunt skill.
Check: The DC is equal to 10 plus the damage received plus the level of the spell that you're trying to cast. The caster receives a -4 penalty to the check if casting within three meters of an enemy.
Use: Automatic

Disable Trap

DESCRIPTION

Ability: Intelligence
Classes: All
Requires Training: Yes
This skill allows the character to perform a variety of actions on a trap.
Check: There are four progressively difficult actions that a character may perform on a trap; the base DC is determined by the difficulty rating of the trap and the difficulty of the action. Disable trap can be used to: examine the trap to determine the difficulty in disarming it (base DC -7), flag the trap so that other party members know to avoid the trap (base DC -5), recover the trap (base DC +10) or disarm it (base DC).
Special: Only rogues may disarm traps with a DC of 25 or greater. With 5 or more ranks in Set Traps a character gains a +2 synergy bonus on Disable Trap checks.
Use: Assess, Flag, Disarm, and Recover are radial menu options off of a detected trap.

Discipline

DESCRIPTION

Ability: Strength
Classes: All
Requires Training: No
A successful check allows the character to resist the effects of any combat feat (Disarm, Called Shot, Sap, or Knockdown).
Check: The DC is equal to the attacker's attack roll.
Use: Automatic

Heal

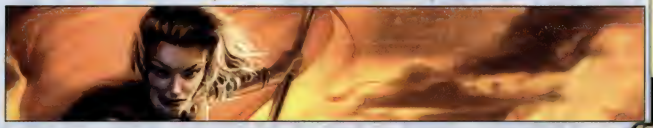
DESCRIPTION

Ability: Wisdom
Classes: All
Requires Training: No
With this skill a character can heal hit points and cure poisons and diseases with a Healing Kit.
Check: Must beat the poison or disease DC. If successful, the target is cured, and is healed with a number of hit points equal to the skill roll, plus all modifiers. If the target suffers from no poisons or diseases, it still is healed of damage.
Use: Use Healing Kit on wounded creature.

Hide

DESCRIPTION

Ability: Dexterity
Classes: All
Requires Training: No
This skill allows a character to hide from enemies.
Check: When hidden, a roll is made against an opposing creature's Spot check, applying any penalties your character might receive from wearing armor. Success means that the opposing creature remains unaware as he passes or your character approaches. Characters may not attempt a Hide check if they are within the line of sight of any intelligent non-party member.
Special: Hide and Move Silently are combined into a single modal Stealth action. Movement in Stealth mode is slower than the normal rate. Wearing armor or using a torch inhibits this ability, but low light can provide a bonus. A character is harder to spot if standing still, and/or if small.
Use: Stealth Mode



Listen

DESCRIPTION

Ability: Wisdom
Classes: All
Requires Training: No
Listen alerts a character to hidden creatures that may be nearby.
Check: Listen detects hidden creatures by opposing their Move Silently check. A successful check renders the hidden creature visible and able to be targeted by the listener.
Special: Standing still provides a +5 bonus to a Listen check. A character with the Alertness feat gains a +2 synergy bonus on Listen checks. Rangers gain a bonus when listening against a favored enemy. Elves, gnomes and halflings gain a +2 racial bonus to Listen checks. Half-elves receive a +1 racial bonus.
Use: Detect Mode

Lore

DESCRIPTION

Ability: Intelligence
Classes: All
Requires Training: No
Lore allows a character to identify unknown magic items.
Check: A roll against an unidentified item's value to determine magical properties that the item may possess.
Special: Bard's are able to identify items easier than other classes. Various spells and items can also give a character a bonus to their lore skill.
Use: Automatic every time the player inspects an item.

Move Silently

DESCRIPTION

Ability: Dexterity
Classes: All
Requires Training: No
A character may sneak quietly past an enemy.
Check: The DC is the opposing creature's Listen check. If you are successful the opposing creature remains unaware as your character moves.
Special: Hide and Move Silently are combined into a single modal Stealth action. Movement in Stealth mode is slower than the normal rate. Wearing armor inhibits this ability, but low light can provide a bonus.
Use: Stealth Mode

Open Lock

DESCRIPTION

Ability: Dexterity
Classes: All
Requires Training: Yes
Using this skill allows the character to gain entrance to locked rooms, or to open locked containers.
Check: The DC is determined by the lock's difficulty rating. A successful check will open the lock.
Special: Thieves' Tools, if used, provide various bonuses to a character's Open Lock attempt, but are destroyed in the attempt whether successful or not.
Use: Select the skill and then target a locked object.

Parry

DESCRIPTION

Ability: Dexterity
Classes: All
Requires Training: No
Parry blocks incoming attacks and occasionally allows for impressive counterattacks.
Check: The DC is the modified attack roll of the incoming blow. A successful parry means that the attack does not damage the parrying character. A character may only parry a number of attacks equal to the number of attacks available to the character.
Special: If the parry is successful and the difference between the roll and the DC is ten points or greater, a counterattack occurs, which is a bonus attack made by the character parrying against the parried opponent.
Use: Select the Parry mode. The character will remain in parry mode until the mode is exited.

Perform

DESCRIPTION

Ability: Charisma
Classes: Bard
Requires Training: No
A Bard using Perform can perform the bardsong, which improves their ally's ability to fight in combat and withstand mind-affecting enemies. Perform is essential for any Bard who wants to have access to their bardsong.
Check: None. The higher the rank in this skill, the better the bardsong.
Use: Select the skill and it will affect a 30-foot radius of allies around the singer.

Persuade

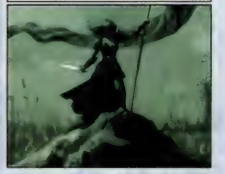
DESCRIPTION

Ability: Charisma
Classes: All
Requires Training: No
In conversation a character has the option to persuade others to reveal additional information about plots or to give bonus treasure.
Check: The DC is determined by the NPC being spoken to.
Use: Used in conversation.

Pick Pocket

DESCRIPTION

Ability: Dexterity
Classes: All
Requires Training: Yes
Pick Pocket allows a character to remove items from another's backpack.
Check: There are two steps to picking pockets. First the item must be acquired, and then the targeted creature must not notice the theft. To steal the item, the base DC from a neutral or tolerant creature is 20, and a hostile creature is 30. This roll is affected by armor check penalties. The targeted creature makes an opposed Spot check vs. the Pick Pocket check of your character. Hostile creatures get a +10 bonus to their Spot checks against Pick Pocket. If the opposed roll succeeds, they have detected your character's attempt to steal. An NPC who detects the attempt will turn hostile, whereas a PC will be informed that you have attempted to pick-pocket them. If, however, both checks succeed for your character, then he or she successful managed to steal the item without being detected.
Use: Select skill, then select valid target.



COMPLETE ENCYCLOPEDIA OF SKILLS!

Spells

Acid Fog

Stats

Caster Level(s): Wizard / Sorcerer 6
Innate Level: 6
School: Conjuration
Descriptor(s): Acid
Component(s): Verbal, Somatic
Range: Long
Area of Effect / Target: Large
Duration: 1 Round / 2 Level
Additional Counter Spells:
Save: Fortitude
Spell Resistance: Yes

Description

Acid Fog creates a thick, clinging, greenish cloud. Creatures inside the cloud must make a Fortitude save or have their movement reduced by 50%. Every round a creature spends in the cloud, it suffers 2d6 acid damage

Animate Dead

Stats

Caster Level(s): Cleric 3, Wizard / Sorcerer 5
Innate Level: 3
School: Necromancy
Descriptor(s): Evil
Component(s): Verbal, Somatic
Range: Short
Area of Effect / Target: Point
Duration: 24 Hours
Additional Counter Spells: Sunbeam
Save: None
Spell Resistance: No

Description

Animate Dead summons forth an undead minion. The type of undead summoned is dependent upon the caster level:
Levels 1 to 5: Tyrantfog zombie
Levels 6 to 9: Skeletal warrior
Level 10 or more: Skeletal knig

Aura versus Alignment

Stats

Caster Level(s): Cleric 8
Innate Level: 8
School: Abjuration
Component(s): Verbal, Somatic
Range: Personal
Area of Effect / Target: Caster
Duration: 1 Round / Level
Additional Counter Spells:
Save: Will, Special
Spell Resistance: Yes

Description

When this spell is cast, the caster chooses to be protected from either good or evil. He receives a +4 deflection bonus to Armor Class, immunity to mind-affecting spells and spell-like abilities used by creatures of the chosen alignment, and spell resistance 25 against spells cast by creatures of the chosen alignment. Creatures of the chosen alignment also take 6 + 1d8 damage each time they successfully strike the Cleric.

Aid

Stats

Caster Level(s): Cleric 2, Paladin 2, Ranger 3
Innate Level: 2
School: Enchantment
Descriptor(s): Mind-Affecting
Component(s): Verbal, Somatic
Range: Touch
Area of Effect / Target: Single
Duration: 1 Turn / Level
Additional Counter Spells: Doom
Save: Harmless
Spell Resistance: No

Description

The target gains a +1 bonus to attack rolls, a +1 bonus on saving throws versus fear, and a 1d8 bonus to maximum hit points.

Aura of Vitality

Stats

Caster Level(s): Druid 7
Innate Level: 7
School: Transmutation
Component(s): Verbal, Somatic
Range: Close
Area of Effect / Target: Colossal
Duration: 1 Round / Level
Additional Counter Spells:
Save: Harmless
Spell Resistance: No

Description

All allies within the area of effect receive a +4 bonus to Strength, Constitution, and Dexterity.

Awaken

Stats

Caster Level(s): Druid 5
Innate Level: 5
School: Transmutation
Component(s): Verbal, Somatic
Range: Touch
Area of Effect / Target: Single
Duration: 24 Hours
Additional Counter Spells:
Save: Will Negates
Spell Resistance: No

Description

The target animal companion gains +4 Strength, +4 Constitution, +3d6 Wisdom, and +2 to attack rolls.

Barkskin

Stats

Caster Level(s): Druid 2
Innate Level: 2
School: Transmutation
Descriptor(s): None
Component(s): Verbal, Somatic
Range: Touch
Area of Effect / Target: Single
Duration: 1 Hour / Level
Additional Counter Spells:
Save: Harmless
Spell Resistance: No

Description

Barkskin hardens the target creature's skin, granting a natural armor bonus to Armor Class based on the caster's level:
Level 1- 6: +3
Level 7-12: +4
Levels 13+: +5

Bless

Stats

Caster Level(s): Cleric 1, Paladin 1
Innate Level: 1
School: Enchantment
Descriptor(s): Mind-Affecting
Component(s): Verbal, Somatic
Range: Long
Area of Effect / Target: Gargantuan
Duration: 1 Turn / Level
Additional Counter Spells: Doom
Save: Harmless
Spell Resistance: No

Description

All allies within the area of effect gain a +1 bonus to attack and damage, along with a +1 on Will saves against fear.

Burning Hands

Stats

Caster Level(s): Wizard / Sorcerer 1
Innate Level: 1
School: Transmutation
Descriptor(s): Fire
Component(s): Verbal, Somatic
Range: 30 ft
Area of Effect / Target: Spell Cone
Duration: Instant
Additional Counter Spells:
Save: Reflex 1/2
Spell Resistance: Yes

Description

A cone of fire shoots forth from the caster's hands, burning all those within its area of effect for 1d4 damage per caster level, up to a maximum of 5d4.

Bestow Curse

Stats

Caster Level(s): Bard 3, Cleric 3, Wizard / Sorcerer 3
Innate Level: 3
School: Transmutation
Component(s): Verbal, Somatic
Range: Touch
Area of Effect / Target: Single
Duration: Permanent
Additional Counter Spells: Remove Curse
Save: Will Negates
Spell Resistance: Yes

Description

Bestow Curse lowers all of the target creature's ability scores by 2.

Blindness / Deafness

Stats

Caster Level(s): Bard 2, Cleric 3, Wizard / Sorcerer 2
Innate Level: 2
School: Enchantment
Component(s): Verbal
Range: Medium
Area of Effect / Target: Single
Duration: 1 Round / Level
Additional Counter Spells: Remove Blindness/Deafness
Save: Fortitude Negates
Spell Resistance: Yes

Description

The target creature is struck blind and deaf.

Call Lightning

Stats

Caster Level(s): Druid 3
Innate Level: 3
School: Evocation
Descriptor(s): Electricity
Component(s): Verbal, Somatic
Range: Long
Area of Effect / Target: Large
Duration: Instant
Additional Counter Spells:
Save: Reflex 1/2
Spell Resistance: Yes

Description

A bolt of lightning strikes the selected target and arcs to all enemies within the area of effect, doing 1d6 points of electrical damage per caster level, to a maximum of 10d6.

Blade Barrier

Stats

Caster Level(s): Cleric 6
Innate Level: 6
School: Evocation
Component(s): Verbal, Somatic
Range: Medium
Area of Effect / Target: Wall 30 ft Long
Duration: 1 Round / Level
Additional Counter Spells:
Save: None
Spell Resistance: Yes

Description

Blade Barrier brings into being a 30-ft long and 3-ft wide wall of stabbing blades. All those passing through the wall take 1d6 points of damage per caster level

Bull's Strength

Stats

Caster Level(s): Bard 2, Cleric 2, Druid 2, Paladin 2, Wizard / Sorcerer 2
Innate Level: 2
School: Transmutation
Component(s): Verbal, Somatic
Range: Touch
Area of Effect / Target: Single
Duration: 1 Hour / Level
Additional Counter Spells: Ray of Enfeeblement
Save: Harmless
Spell Resistance: No

Description

The target creature's Strength is increased by 1d4+1.

Cat's Grace

Stats

Caster Level(s): Bard 2, Ranger 2, Wizard / Sorcerer 2
Innate Level: 2
School: Transmutation
Component(s): Verbal, Somatic
Range: Touch
Area of Effect / Target: Single
Duration: 1 Hour / Level
Additional Counter Spells:
Save: Harmless
Spell Resistance: No

Description

The target creature's Dexterity is increased by 1d4+1.

MASTER SPELL BOOK FOR DISCERNING CASTERS!

Weapons

Acid Arrow
Weapon Type: Arrow
Properties
Damage Bonus: Acid 1d6 Damage
Damage	Cost
N/A	2134

Acid Dart
Weapon Type: Dart
Properties
Damage Bonus: Acid 1d4 Damage
Enhancement Bonus +1
Damage Type: Piercing
Damage	Cost
1d4	1574

Aiedo Wither-Stick
Weapon Type: Quarterstaff
Properties
Enhancement Bonus +1
On Hit: Ability: Drain Strength
Damage Type: Bludgeoning
Damage	Cost
1d6	23122

Angurvadal (Flame Tongue)
Weapon Type: Longsword
Properties
Damage Bonus: Fire 1d6 Damage
Enhancement Bonus +2
Damage Type: Slashing and Piercing
Damage	Cost
1d8	21808

Arc Asunder
Weapon Type: Halberd
Properties
Damage Bonus: Electrical 1d6 Damage
Enhancement Bonus +1
Light Low (10m) Blue
Skill Bonus: Parry +4
Damage Type: Slashing and Piercing
Damage	Cost
1d10	18992

Ardulia's Fall
Weapon Type: Dire Mace
Properties
Enhancement Bonus +2
On Hit: Slow 25% / 3 Rounds
Damage Type: Bludgeoning
Damage	Cost
1d8	16900

Arrow
Weapon Type: Arrow
Properties
N/A
Damage	Cost
N/A	1

Arrow of the Vampire
Weapon Type: Arrow
Properties
Regeneration: Vampiric +2
Damage	Cost
N/A	983

Ascorian Double Axe
Weapon Type: Double Axe
Properties
Enhancement Bonus vs. Alignment Group: Evil +1
Regeneration: Vampiric +3
Damage Type: Slashing and Piercing
Damage	Cost
1d8	7980

Asp Flail
Weapon Type: Heavy Flail
Properties
Damage Bonus: Acid 1d6 Damage
Enhancement Bonus +1
Damage Type: Bludgeoning
Damage	Cost
1d10	10608

Asp's Nest
Weapon Type: Dart
Properties
Enhancement Bonus +1
On Hit: Poison 1d2 Constitution Damage
Damage Type: Piercing
Damage	Cost
1d4	4204

Astral Blade +1
Weapon Type: Longsword
Properties
Damage Bonus: Sonic 1d4 Damage
Enhancement Bonus +1
Damage Type: Slashing and Piercing
Damage	Cost
1d8	6330

Astral Blade +2
Weapon Type: Longsword
Properties
Damage Bonus: Sonic 1d10 Damage
Enhancement Bonus +2
Damage Type: Slashing and Piercing
Damage	Cost
1d8	40980

Axe for the Tall Kin +1
Weapon Type: Double Axe
Properties
Enhancement Bonus +1
Light Low (10m) White
Damage Type: Slashing and Piercing
Damage	Cost
1d8	2940

Axe for the Tall Kin +3
Weapon Type: Double Axe
Properties
Enhancement Bonus +3
Light Low (10m) White
Damage Type: Slashing and Piercing
Damage	Cost
1d8	20540

Axe of Dead Illusk
Weapon Type: Battleaxe
Properties
Enhancement Bonus +2
On Hit: Fear 50% / 2 Rounds
Regeneration: Vampiric +3
Damage Type: Slashing
Damage	Cost
1d8	37864

Axe of the Coming Storm
Weapon Type: Throwing Axe
Properties
Enhancement Bonus +3
Damage Bonus: Electrical 1d4 Damage
Damage Type: Slashing
Damage	Cost
1d6	2702

Axe of the Culling
Weapon Type: Great Axe
Properties
Damage Bonus: Fire 1d6 Damage
Enhancement Bonus +3
On Hit: Dispel Magic
Damage Type: Slashing
Damage	Cost
1d12	121720

Bastard Sword
Weapon Type: Bastard Sword
Properties
N/A
Damage Type: Slashing and Piercing
Damage	Cost
1d10	70

Bastard Sword +1
Weapon Type: Bastard Sword
Properties
Enhancement Bonus +1
Damage Type: Slashing and Piercing
Damage	Cost
1d10	1688

Bastard Sword +2
Weapon Type: Bastard Sword
Properties
Enhancement Bonus +2
Damage Type: Slashing and Piercing
Damage	Cost
1d10	7288

Bastard Sword +3
Weapon Type: Bastard Sword
Properties
Enhancement Bonus +3
Damage Type: Slashing and Piercing
Damage	Cost
1d10	16890

Battleaxe
Weapon Type: Battleaxe
Properties
N/A
Damage Type: Slashing
Damage	Cost
1d8	20

Battleaxe +1
Weapon Type: Battleaxe
Properties
Enhancement Bonus +1
Damage Type: Slashing
Damage	Cost
1d8	1638

Battleaxe +2
Weapon Type: Battleaxe
Properties
Enhancement Bonus +2
Damage Type: Slashing
Damage	Cost
1d8	7238

DAMAGE, SPECIAL PROPERTIES, AND PRICE FOR EVERY WEAPON IN THE GAME!

Armor

Adamantine Breastplate
Armor Type: Armor (Type 5)
Properties
AC Bonus +2
Damage Reduction: +1 Soak 5 Damage
AC: 5 | Cost: 17874

Adept's Tunic
Armor Type: Armor (Type 0)
Properties
N/A
AC: 0 | Cost: 1

Archer's Belt
Armor Type: Belt
Properties
Immunity: Damage Type: Piercing 10% Immunity Bonus
AC: 0 | Cost: 1563

Armor of Comfort
Armor Type: Armor (Type 6)
Properties
AC Bonus +2
Base Item Weight Reduction 20% of Weight
Skill Bonus: Heal +2
AC: 6 | Cost: 6202

Armor of Command
Armor Type: Armor (Type 4)
Properties
Ability Bonus: Charisma +2
AC Bonus +3
Skill Bonus: Persuade +5
AC: 4 | Cost: 31572

Armor of Fleetness
Armor Type: Armor (Type 1)
Properties
AC Bonus +3
Haste
AC: 1 | Cost: 37337

Armor of Freedom
Armor Type: Armor (Type 8)
Properties
AC Bonus +3
Freedom of Movement
AC: 8 | Cost: 22752

Armor of Horus-Re
Armor Type: Armor (Type 4)
Properties
AC Bonus +3
Damage Resistance: Electrical Resist 10 / -
Damage Resistance: Sonic Resist 10 / -
AC: 4 | Cost: 16381

Armor of Loyalty
Armor Type: Armor (Type 3)
Properties
AC Bonus +1
Cast Spell: Charm Person (2) 3 Uses/Day
Immunity: Miscellaneous: Mind-Affecting Spells
AC: 3 | Cost: 4630

Armor of the Stars
Armor Type: Armor (Type 1)
Properties
AC Bonus +3
Immunity: Miscellaneous: Disease
Light Low (10m) Blue
AC: 1 | Cost: 15391

Armor of the Wolf
Armor Type: Armor (Type 1)
Properties
AC Bonus +3
AC Bonus vs. Racial Group: Shapechanger +3
Skill Bonus: Move Silently +2
AC: 1 | Cost: 18495

Armor of Thorns
Armor Type: Armor (Type 3)
Properties
AC Bonus +2
Cast Spell: Vampiric Touch (5) 1 Use/Day
AC: 3 | Cost: 7439

Aslyferund Elven Chain
Armor Type: Armor (Type 1)
Properties
AC Bonus +3
Damage Reduction: +1 Soak 5
Damage
Use Limitation: Racial Type: Elf
Use Limitation: Racial Type: Half-Elf
AC: 1 | Cost: 26117

Assassin's Garb
Armor Type: Armor (Type 0)
Properties
N/A
AC: 0 | Cost: 1

Aurumvorax Armor
Armor Type: Armor (Type 2)
Properties
AC Bonus +2
Damage Resistance: Piercing Resist 5 / -
Damage Resistance: Slashing Resist 5 / -
AC: 2 | Cost: 12862

Balduran's Armor
Armor Type: Armor (Type 7)
Properties
Ability Bonus: Charisma +2
AC Bonus +2
Regeneration +2
AC: 7 | Cost: 25951

Banded Mail
Armor Type: Armor (Type 6)
Properties
N/A
AC: 6 | Cost: 200

Banded Mail +1
Armor Type: Armor (Type 6)
Properties
AC Bonus +1
AC: 6 | Cost: 856

Banded Mail +2
Armor Type: Armor (Type 6)
Properties
AC Bonus +2
AC: 6 | Cost: 3124

Banded Mail +3
Armor Type: Armor (Type 6)
Properties
AC Bonus +3
AC: 6 | Cost: 7012

Barbarian's Outfit
Armor Type: Armor (Type 0)
Properties
N/A
AC: 0 | Cost: 1

Bard's Tunic
Armor Type: Armor (Type 0)
Properties
N/A
AC: 0 | Cost: 1

Belt of Fire Giant Strength
Armor Type: Belt
Properties
Ability Bonus: Strength +5
AC: 0 | Cost: 36001

Belt of Frost Giant Strength
Armor Type: Belt
Properties
Ability Bonus: Strength +4
AC: 0 | Cost: 23041

Belt of Hill Giant Strength
Armor Type: Belt
Properties
Ability Bonus: Strength +3
AC: 0 | Cost: 12961

Belt of Inertial Barrier
Armor Type: Belt
Properties
AC Bonus +1
Damage Resistance: Bludgeoning: Resist 5 / -
Damage Resistance: Slashing: Resist 5 / -
Damage Vulnerability: Piercing 25%
Damage Vulnerability
AC: 0 | Cost: 6960

Belt of the Performer +1
Armor Type: Belt
Properties
Ability Bonus: Charisma +1
Skill Bonus: Perform +1
Skill Bonus: Persuade +1
AC: 0 | Cost: 2074

Belt of the Performer +3
Armor Type: Belt
Properties
Ability Bonus: Charisma +3
Skill Bonus: Perform +2
Skill Bonus: Persuade +2
AC: 0 | Cost: 16647

Belt of the Performer +5
Armor Type: Belt
Properties
Ability Bonus: Charisma +5
Skill Bonus: Perform +5
Skill Bonus: Persuade +5
AC: 0 | Cost: 48442

Benzo's Luck
Armor Type: Armor (Type 1)
Properties
AC Bonus +1
AC Bonus vs. Alignment Group: Chaotic +3
AC Bonus vs. Alignment Group: Evil +3
AC: 1 | Cost: 13769

Black Flame Armor
Armor Type: Armor (Type 3)
Properties
AC Bonus +5
Damage Resistance: Fire Resist 30 / -
AC: 3 | Cost: 36436

Black Robe of the Archmagi
Armor Type: Armor (Type 0)
Properties
AC Bonus +5
Bonus Feat: Spell Penetration
Saving Throw Bonus: Universal +1
Spell Resistance Bonus of 10 SR
Use Limitation: Alignment Group: Evil
Use Limitation: Class: Sorcerer
Use Limitation: Class: Wizard
AC: 0 | Cost: 48442

AC RATING, SPECIAL PROPERTIES, AND PRICE FOR EVERY PIECE OF ARMOR!

Items

Ale
PROPERTIES

Cast Spell: Alcohol, Beer Single Use
COST 2

Alexandrite
PROPERTIES

N/A
COST 145

Amethyst
PROPERTIES

N/A
COST 40

Amulet of Acid Resistance
PROPERTIES

Damage Resistance: Acid Resist 20 / -
COST 621

Amulet of Cold Resistance
PROPERTIES

Damage Resistance: Cold Resist 20 / -
COST 811

Amulet of Health
PROPERTIES

Immunity: Miscellaneous: Disease
Immunity: Miscellaneous: Level/Ability Drain
Immunity: Miscellaneous: Poison
COST 12251

Amulet of Electrical Resistance
PROPERTIES

Damage Resistance: Electrical Resist 20 / -
COST 1026

Amulet of Fire Resistance
PROPERTIES

Damage Resistance: Fire Resist 20 / -
COST 1266

Amulet of Natural Armor +1
PROPERTIES

AC Bonus +1
COST 657

Amulet of Natural Armor +2
PROPERTIES

AC Bonus +2
COST 2925

Amulet of Natural Armor +3
PROPERTIES

AC Bonus +3
COST 6813

Amulet of Natural Armor +4
PROPERTIES

AC Bonus +4
COST 12321

Amulet of Natural Armor +5
PROPERTIES

AC Bonus +5
COST 19449

Amulet of Power
PROPERTIES

Bonus Feat: Spell Penetration
Bonus Spell Slot of Level: Wizard Level 3
Bonus Spell Slot of Level: Wizard Level 4
Bonus Spell Slot of Level: Wizard Level 5
Use Limitation: Class: Wizard
COST 19141

Amulet of the Long Death +1
PROPERTIES

Ability Bonus: Constitution +1
Spell Resistance Bonus of 10 SR
COST 3516

Amulet of the Long Death +3
PROPERTIES

Ability Bonus: Constitution +3
Spell Resistance Bonus of 10 SR
COST 18276

Amulet of the Master
PROPERTIES

Skill Bonus: Disable Trap +6
Skill Bonus: Lore +6
Skill Bonus: Open Lock +6
Skill Bonus: Persuade +6
Skill Bonus: Search +6
Spell Resistance 20
COST 44356

Amulet of the Red Tiger Tribe +1
PROPERTIES

Ability Bonus: Strength +1
Immunity: Miscellaneous: Sneak Attack
COST 4841

Amulet of the Red Tiger Tribe +3
PROPERTIES

Ability Bonus: Strength +3
Immunity: Miscellaneous: Sneak Attack
COST 21161

Amulet of the Silent Lord +4
PROPERTIES

Ability Bonus: Constitution +5
Immunity: Miscellaneous: Level/Ability Drain
Spell Resistance Bonus of 10 SR
COST 48651

Amulet of the Uthgardt +4
PROPERTIES

Ability Bonus: Strength +5
Freedom of Movement
Immunity: Miscellaneous: Sneak Attack
COST 44891

Amulet of Will +1
PROPERTIES

Saving Throw Bonus: Specific: Will +1
COST 343

Amulet of Will +2
PROPERTIES

Saving Throw Bonus: Specific: Will +2
COST 1526

Amulet of Will +3
PROPERTIES

Saving Throw Bonus: Specific: Will +3
COST 3554

Amulet of Will +4
PROPERTIES

Saving Throw Bonus: Specific: Will +4
COST 6427

Amulet of Will +5
PROPERTIES

Saving Throw Bonus: Specific: Will +5
COST 10145

Aventurine
PROPERTIES

N/A
COST 20

Average Acid Splash Trap Kit
PROPERTIES

Trap: Average Acid Splash
COST 79

Average Blob of Acid Trap Kit
PROPERTIES

Trap: Average Blob of Acid
COST 490

Average Electrical Trap Kit
PROPERTIES

Trap: Average Electrical
COST 1588

Average Fire Trap Kit
PROPERTIES

Trap: Average Fire
COST 961

Average Frost Trap Kit
PROPERTIES

Trap: Average Frost
COST 79

Average Gas Trap Kit
PROPERTIES

Trap: Average Gas
COST 3063

Average Holy Trap Kit
PROPERTIES

Trap: Average Holy
COST 123

Average Negative Trap Kit
PROPERTIES

Trap: Average Negative
COST 123

Average Sonic Trap Kit
PROPERTIES

Trap: Average Sonic
COST 79

Average Spike Trap Kit
PROPERTIES

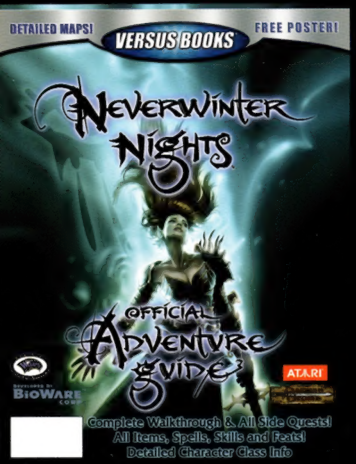
Trap: Average Spike
COST 20

Average Tangle Trap Kit
PROPERTIES

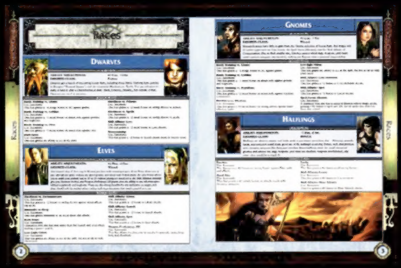
Trap: Average Tangle
COST 241

DESCRIPTION AND PRICE FOR ALL ITEMS!

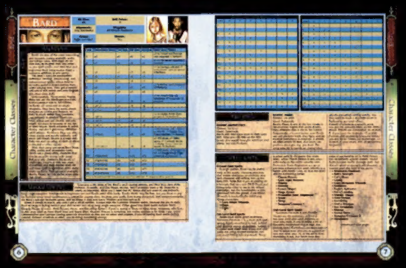
Get BOTH Guides to Completely Cover ALL of Neverwinter Nights!



THE ONLY GUIDE THAT COMPLETELY COVERS ALL QUESTS AND ITEMS.

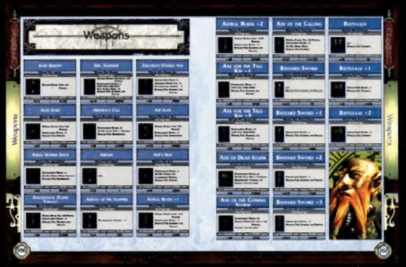


STATISTICS, ANALYSIS & BONUS FEAT LISTINGS FOR EVERY RACE!



ANALYSIS, LEVEL PROGRESSION CHARTS, AND STATS FOR ALL CLASSES!

COMPLETE WALKTHROUGH INCLUDING ALL SIDE-QUESTS!



EVERY ITEM, WEAPON, ARMOR, AND SPELL REVEALED!

Order Now Or Go Online For Other Great Titles From VERSUS BOOKS.

ORDER FORM — VERSUS BOOKS — ORDER FORM

copies	name of book	price ea.	subtotal
_____	NWN Adventure Guide	@ 19.99	_____
_____	Sonic Adventure 2/Sonic ADV	@ 14.99	_____
_____	Grandia II	@ 14.99	_____
_____	Maximo	@ 14.99	_____
_____	Half-Life(PS2)	@ 6.99	_____
_____	Luigi's Mansion	@ 6.99	_____
_____	Gran Turismo 3	@ 6.99	_____
_____	Pokémon Crystal	@ 6.99	_____
_____	Devil May Cry	@ 6.99	_____
_____	BG2: Throne of Bhaal	@ 6.99	_____
	Shipping charges: $4 for one book, 50¢ for each additional book		_____
	TOTAL: (add $4 per copy if ordering from outside U.S.)		_____

Name: _____

Address: _____

City: _____ State: _____ Zip: _____

email: _____

CREDIT CARD ORDERS ONLY

Name on Card: _____

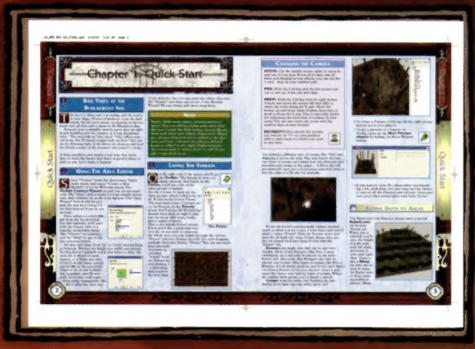
Credit Card #: _____

Visa or MC: _____ Expiration: _____

Send to: Versus Books, PMB#404,
2532 Santa Clara, CA 94501-4634
photocopies of order form accepted
for faster service visit our website:

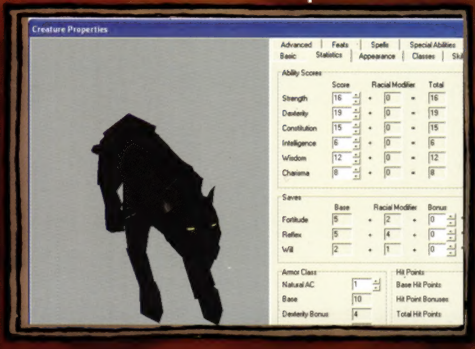
www.versusbooks.com

THE OFFICIAL WORLD BUILDER GUIDE



The most detailed documentation available on the *NWN* toolset -- create a world of near-infinite possibilities.

A step-by-step guide to creating the scripting that runs your world. Includes a full appendix on C programming.



A bestiary including full monster data, strategy, and society notes to make your campaign as realistic as possible.

Examples from our own module serve as a side-by-side comparison with our in-depth world-building tutorial.

ALSO AVAILABLE: THE VERSUS BOOKS ADVENTURE GUIDE!

FROM VERSUS BOOKS™

ISBN 1-931886-13-X

NEVERWINTER NIGHTS © 2002 Infogrames Entertainment, S.A. All Rights Reserved. Manufactured and marketed by Infogrames, Inc., New York, NY. Portions © 2002 Bioware Corp. BioWare Aurora Engine copyright 1997-2002 BioWare Corp. All Rights Reserved.BioWare, the BioWare Aurora Engine, and the BioWare Logo are trademarks of BioWare Corp. All Rights Reserved. Neverwinter Nights, Forgotten Realms, the Forgotten Realms logo, Dungeons & Dragons logo, Dungeon Master, D&D, and the Wizards of the Coast logo are trademarks owned by Wizards of the Coast, Inc., a subsidiary of Hasbro, Inc. and are used by Infogrames Entertainment, S.A. under license. All Rights Reserved. Windows and Windows 95/98/2000 are registered trademarks of Microsoft Corporation. All Rights Reserved. The ratings icon is a trademark of the Interactive Digital Software Association. All other trademarks are the property of their respective owners.